COME PROGRAMMARE IL TUO COMMODORE 64

Indispensabile per chi non ha mai programmato in BASIC e vorrebbe farlo in pochissimo tempo.

di TIM HARTNELL e ROBERT YOUNG

COME PROGRAMMARE IL TUO COMMODORE 64

Indispensabile per chi non ha mai programmato in BASIC e vorrebbe farlo in pochissimo tempo.

di TIM HARTNELL e ROBERT YOUNG

JACOPO CASTELFRANCHI EDITORE
Via dei Lavoratori, 124
CINISELLO BALSAMO (MI)

Prima edizione americana : Maggio 1984

Prima edizione italiana : Agosto 1984

Stampato in Italia da
Gemm Grafica S.r.l.
Via Magretti - Paderno Dugnano (MI)

I N D I C E

P R E F A Z I O N E

Se non hai mai programmato un computer prima d'ora, e ti piacerebbe essere capace di programmare il COMMODORE 64 in poco tempo, allora questo libro e' stato scritto per te.

Penso che tu abbia comprato molti, se non tutti i programmi che si inseriscono nel COMMODORE 64.

Malgrado cio', ti sei probabilmente accorto che sarebbe utile avere almeno una discreta conoscenza su come programmare un COMMODORE 64, per impressionare i tuoi amici, insegnare ai tuoi bambini e creare programmi per tua utilita' o per diletto. questo libro ti mostrera' come fare cio' nel minor tempo possibile.

Troverai le parole piu' importanti per l'uso del COMMODORE 64 e avrai una collezione di interessanti programmi a cui potrai ricorrere ogni volta che ne avrai bisogno.

Un grazie a Kelvin Benson del Southern Office Equipment e a Gary Landers del Geelong Computer Centre per il loro

aiuto e i loro consigli durante la stesura di questo libro.

Robert Young
Geelong, 1983

C A P I T O L O   1

I L   C O M M O D O R E   64

Hai appena comprato un grande computer e in questo libro ti mostreremo come usarlo facilmente e velocemente. Non ti preoccupare se questa e' la prima volta che sei davanti ad un computer; procederemo lentamente e per gradi in modo che tu non abbia problemi nell'apprendere.

LEGGERE IL LIBRO CON IL COMPUTER ACCESO

E' fondamentale che tu abbia il computer acceso tutte le volte che leggerai questo libro. Questo e' un "libro d'azione", e diversamente dai romanzi, non e' stato pensato semplicemente per essere letto. Questo libro e' come un testo contenente le istruzioni per imparare a guidare. Non potrai mai imparare a cambiare le marce o a maneggiare un'auto nel traffico, senza prendere in mano il volante. Cosi' deve essere con il computer e questo libro. Prova un nuovo comando solo quando ti viene spiegato, esercitati con i giochi e gli altri programmi, e ti accorgerai che stai facilmente imparando a programmare.

LA TASTIERA

Prima di tutto devi saperti districare con i tasti. La tastiera impaurisce quasi la prima volta che la si osserva, ma una volta usata per un'ora, sarai sorpreso di come ti sia diventata familiare.

Una volta acceso il COMMODORE 64, lo schermo ti si presentera' di colore blu scuro con il bordo azzurro. Su di esso saranno impresse queste parole "COMMODORE 64 BASIC V2 64K RAM SYSTEM 38911 BASIC BYTES FREE". Sotto comparira' la parola READY. Questo e' il modo in cui il computer ti comunica che e' pronto per il funzionamento. Questo messaggio apparira' ogni volta che il computer accomplira'

una funzione e sara' pronto per la prossima. Sotto detto messaggio dovrebbe esserci un quadratino lampeggiante: il cursore. Esso mostra la tua posizione sullo schermo quando stai scrivendo comandi o informazioni. La lettera o il carattere che tu inserirai di seguito apparira' nella posizione attuale del cursore che allora si muovera' di uno spazio preparandosi per il prossimo carattere. La tastiera funziona proprio come una macchina da scrivere. Contiene le lettere dell'alfabeto, i numeri da 0 a 9, i simboli matematici ed altri simboli. come puoi notare, molti tasti hanno dei simboli grafici sulla loro facciata frontale. Parleremo di cio' piu' avanti. In fondo alla tastiera si trova la barra per gli spazi. Ai lati della tastiera si trovano altri tasti, ma per ora ci occuperemo soltanto dei tasti con le lettere dell'alfabeto.

Il COMMODORE 64 ha due sistemi di scrittura: il sistema con lettere maiuscole e caratteri grafici e quello con lettere minuscole. Una volta acceso il computer, il sistema che ti si presenta e' quello con lettere maiuscole e caratteri grafici. Questo significa che tutto l'alfabeto apparira' maiuscolo e che si possono usare i simboli grafici presenti sulle facciate frontali dei tasti.

Cominciamo. Usando il computer come una macchina da scrivere, scrivi il tuo nome. Questo apparira' sullo schermo sotto la scritta READY, con il cursore lampeggiante dopo l'ultima lettera. Premi qualche volta la barra per gli spazi e osserverai il cursore muoversi verso destra. Ora scrivi ancora il tuo nome. Ti sarai accorto che lo schermo del tuo computer puo' essere usato proprio come un foglio di carta in una macchina da scrivere.

IL TASTO SHIFT

Nella grafia a caratteri maiuscoli e simboli grafici il tasto SHIFT ha due funzioni. Congiuntamente ai tasti con due simboli sulla facciata superiore, ad esempio i tasti con i numeri o quelli con la punteggiatura, il tasto SHIFT ti permette di far apparire il secondo simbolo o simbolo superiore. Per provare cio' premi il tasto SHIFT e tutti i tasti dei numeri da 1 a 9. Ora premili ancora senza premere il tasto SHIFT.

La seconda funzione del tasto SHIFT nella grafia a caratteri maiuscoli e simboli grafici, e' quella di renderti capace di usare alcuni simboli grafici presenti sulla facciata frontale di alcuni tasti. Per esempio premi il tasto SHIFT e il tasto Q. Comparira' sullo schermo un cerchio pieno. Si tratta del carattere grafico presente sulla parte destra della facciata frontale del tasto Q. Prova con altri tasti. Verificherai cosi' che il tasto SHIFT serve a darti il simbolo grafico presente sulla parte destra della facciata frontale dei tasti. Per quanto riguarda i simboli grafici presenti sulla parte sinistra della facciata frontale dei tasti, dovremo occuparci del tasto COMMODORE.

IL TASTO COMMODORE

Si tratta del tasto con impresso il simbolo del COMMODORE che si trova nell'angolo inferiore sinistro della tastiera. Anch'esso ha svariati usi. Premi ancora il tasto Q, ma questa volta premi contemporaneamente il tasto COMMODORE. Comparira' sullo schermo il simbolo grafico presente sulla parte sinistra della facciata frontale del tasto Q. Prova a scrivere alcuni di
questi simboli usando sia il tasto SHIFT, sia il tasto COMMODORE fino a che l'operazione ti sia diventata familiare.

Se il tasto COMMODORE e il tasto SHIFT vengono premuti contemporaneamente, il computer modifica il sistema di grafia usato, nel sistema con lettere minuscole. Prova subito. Constaterai che tutto cio' che hai finora scritto sullo schermo apparira' ora a lettere minuscole. Prova a scrivere il tuo nome e osserva come appare ora in lettere minuscole. Per avere le lettere maiuscole, bisogna usare il tasto SHIFT esattamente come se si trattasse di una macchina da scrivere. Per riportare il computer alla grafia con lettere maiuscole e simboli grafici premi ancora il taso SHIFT e il tasto COMMODORE contemporaneamente.

Il tasto COMMODORE ha anche un'altro uso, ma lo vedremo in seguito.

IL TASTO CONTROL

Questo e' il tasto contrassegnato dalla sigla CTRL, che si trova sul lato sinistro della tastiera. Questo tasto serve ad evidenziare i colori le cui sigle si trovano sulla facciata frontale dei tasti dei numeri da 1 a 8. Premi contemporaneamente il tasto CTRL e il tasto con il numero 1. ora scrivi il tuo nome (o qualcos'altro se ti sei stancato di scrivere sempre le stesse cose). Questa volta le parole sullo schermo appariranno in nero. Ora premi il tasto CTRL con uno degli altri tasti con i colori. Hai a tua disposizione otto differenti colori, usando il tasto CTRL, e altri otto usando il tasto COMMODORE. Ecco una lista dei colori e dei rispettivi tasti.

KEY	COLOR
CTRL 1	NERO
CTRL 2	BIANCO
CTRL 3	ROSSO
CTRL 4	TURCHESE
CTRL 5	PORPORA
CTRL 6	VERDE
CTRL 7	BLU
CTRL 8	GIALLO
COMM 1	ARANCIONE
COMM 2	MARRONE
COMM 3	ROSSO CHIARO
COMM 4	GRIGIO 1
COMM 5	GRIGIO 2
COMM 6	VERDE CHIARO
COMM 7	AZZURRO
COMM 8	GRIGIO 3

Provate ad usare questi 16 colori, premendo di volta in volta sia il tasto CTRL, sia quello COMM fino a che l'operazione vi sia diventata familiare.

I tasti 9 e 0 assolvono un'interessante funzione se usati con il tasto CTRL. Premete CTRL e contemporaneamente schiacciate il tasto col numero 9 seguito dal tasto col numero 1. Ora scrivete qualcosa sullo schermo. Il tasto 9,

che porta la segnatura RVS ON (che sta per REVERSE ON) sulla sua facciata frontale, ha invertito i colori e stampato lettere blu su fondo nero. Premete la barra spaziatrice per qualche secondo in modo tale da poter vedere come appare uno spazio invertito. Schiacciando il tasto recante la segnatura RVS OFF, sempre contemporaneamente al tasto CTRL, il computer torna alla normalita'.

IL TASTO RUN/STOP

Premendo il tasto STOP si fermera' non solo l'esecuzione di un programma ma anche il listato. Se questo tasto viene premuto contemporaneamente al tasto RESTORE, il computer tornera' alla condizione in cui era quando e' stato acceso, cioe' uno schermo blu scuro con un bordo azzurro (potete anche cambiare i colori dello schermo e dello sfondo; cio' vi sara' mostrato in un prossimo capitolo). Il computer tornera' anche alla grafia con lettere maiuscole e caratteri grafici e lo stampato ritornera' al suo originale colore. Comunque, se avete inserito un programma in memoria non verra' cancellato.

Se premete il tasto RUN/STOP con il tasto SHIFT per avere RUN, cio' vi permettera' di inserire un programma da cassetta alla memoria del computer.

IL TASTO CTRL/HOME

La funzione di questo tasto, senza premere SHIFT nello stesso tempo, e' HOME. Il cursore tornera' all'angolo superiore sinistro dello schermo, da qualsiasi posizione si trovasse prima.

La funzione del tasto premuto contemporaneamente a SHIFT e' CLR che significa pulire lo schermo. Questa funzione fa in modo che tutti i caratteri presenti sullo schermo scompaiano e che il cursore ritorni all'angolo superiore sinistro. Questo comando non influenza alcun programma che, in quel dato momento, si trovi nella memoria del computer.

IL TASTO INSERT/DELETE

E' il tasto che si trova nell'angolo superiore destro della tastiera. Viene usato per correggere errori nelle righe del programma. Ti parleremo di questo tasto in un capitolo a seguire.

I TASTI CURSOR

I due tasti nell'angolo inferiore destro della tastiera permettono di muovere il cursore sullo schermo. Provate tutte le direzioni. Ricordati di usare il tasto SHIFT per muovere il cursore verso l'alto e verso sinistra. Questi due tasti sono veramente utili per scrivere e li tratteremo ancora nella nostra sezione sull'editing.

IL TASTO RETURN

Questo tasto viene usato per inserire informazioni nella memoria del computer. Ne parleremo nel prossimo capitolo.

I TASTI FUNCTION

I quattro tasti alla destra della tastiera sono chiamati tasti function. Essi non hanno una funzione prestabilita ma sono proggammabili. Cio' significa che li potete usare in un programma per assolvere compiti ripetitivi, per sostituire una serie di tasti o per inserire un set programmato di istruzioni.

Bene, ora che hai familiarizzato con la tastiera e le sue operazioni, continuiamo a scoprire il nostro COMMODORE 64.

C A P I T O L O 2

C O M E S T A M P A R E S U L L O S C H E R M O

Cominciamo ad imparare a programmare, con il comando piu' comunemente usato in BASIC, la parola PRINT.

Stampa cio' che segue sul tuo computer:

```
PRINT 2
```

Il computer non fara' niente fino a che tu premerai il tasto RETURN. Piu' specificatamente, a questo punto esso ignorera' il comando PRINT 2. Ora premi RETURN e vedrai apparire il numero 2 sotto il comando PRINT 2. Questo e' il modo con il quale opera PRINT. Esso riceve l'informazione che segue il comando PRINT (ci saranno alcune eccezioni che apprenderemo piu' avanti) e la stampa sullo schermo.

Se la parola PRINT e' seguita da una somma, il computer operera' l'addizione prima di stampare e ti dara' il risultato. Prova ora. Inserisci il seguente ordine e poi premi RETURN:

```
PRINT 5+3
```

Vedrai apparire il numero 8. Il computer ha sommato 5 e 3, come dall'istruzione ricevuta dal segno (+), poi ne ha stampato il risultato sullo schermo. Esso puo' fare nello stesso modo le sottrazioni. Batti questo e premi RETURN:

```
PRINT 7-2
```

Il computer puo' svolgere un vasto numero di operazioni matematiche, molte delle quali assai piu' sofisticate di una semplice addizione o sottrazione. Ma per cio' che concerne la moltiplicazione il computer non usa il simbolo X bensi' un asterisco (*) e, per la divisione, una sbarra (/).

Il computer non si limita ad una singola operazione. Puoi combinarne quante ne vuoi. Prova il prossimo comando, che combina una moltiplicazione e una divisione, battilo, poi premi RETURN:

PRINT 5*3/2

Sembra abbastanza semplice. Si batte la parola PRINT, di seguito l'informazione che si vuole che il computer stampi, e questo e' tutto.

Quando pero' si operano calcoli matematici sul tuo computer devi tener presente l'ordine con il quale esso svolge le operazioni. Per esempio sommiamo 40 e 10 e dividiamo il risultato per 5:

PRINT 40+10/5

e premiamo RETURN.

Bene, la risposta non sara' assolutamente quella voluta. Il computer svolge le moltiplicazioni e le divisioni prima delle addizioni e sottrazioni. Nel caso precedente ha diviso prima 10 per 5 e poi ha aggiunto 40. Per ovviare a questo problema, bisogna includere la parte del calcolo che si vuole sia fatta per prima, nelle parentesi. Il COMMODORE 64 svolgera' innanzitutto la funzione che si trova tra parentesi. Se c'e' una serie di parentesi all'interno di un'altra serie, esso eseguira' per prime le operazioni contenute nelle parentesi piu' interne. Ma vedremo calcoli di questo tipo piu' avanti.

Ora torniamo al nostro problema originale per avere il risultato richiesto:

PRINT (40+10)/5

Questa volta la risposta e' corretta, 10.

L'ordine con cui il computer svolge le funzioni matematiche e' il seguente:

PRIMO tutto cio' che si trova nelle parentesi
SECONDO numeri negativi
TERZO elevamenti a potenza

QUARTO   moltiplicazioni o divisioni
QUINTO   addizioni o sottrazioni

Quando si trova davanti a piu' operazioni che hanno lo stesso ordine di precedenza, il computer le svolgera' partendo da sinistra verso destra.

Dopo aver operato con i numeri, esaminiamo ora come si stampano lettere e parole. Cominciamo col stampare la linea seguente:

PRINT PROVARE

Al posto della parola 'provare' comparira' uno zero. Il computer ha pensato che volessimo una variabile al posto della parola 'provare' (ti spiegheremo piu' avanti il significato della parola 'variabile'). Piu' semplicemente esso ha pensato volessimo stampare un numero che avesse il nome 'provare'.

STRINGHE

Se invece vuoi che il computer stampi la parola 'provare' devi inserire la stessa tra virgolette, cosi':

PRINT "PROVARE"

Questa volta quando premi RETURN avrai finalmente la parola 'provare' sullo schermo. Vale la pena di ricordarselo. Quando vuoi che il computer stampi delle parole o combinazioni di parole, simboli, spazi e numeri, e' necessario mettere le virgolette alla serie di caratteri che vuoi stampare. In gergo, l' informazione inclusa tra virgolette, si chiama 'stringa'.Potresti, volendo, mettere solo la prima virgoletta: il risultato sarebbe il medesimo.

UN PRIMO PROGRAMMA

Batti la seguente frase nel tuo computer. Noterai che ogni riga inizia con un numero:

```
10 PRINT"GIANNI E MARIA"
```

Batti le righe seguenti (20,30,40) e premi RETURN ogni volta che hai terminato una riga:

```
20 PRINT"SALIRONO SULLA COLLINA"
30 PRINT"A PRENDERE UN SECCHIO"
40 PRINT"D' ACQUA"
```

Quando esegui questo programma vedrai sullo schermo la seguente frase:

```
SALIRONO SULLA COLLINA
A PRENDERE UN SECCHIO D'ACQUA
```

PER AGGIUNGERE NUOVE LINEE

Il computer ti permette di inserire le righe, qualsiasi ordine tu scelga. Sebbene il tuo primo programma e molti altri di questo testo siano numerati, partendo da 10, con salti di 10, non c'e' ragione particolare per cui tu debba seguire questa numerazione. Comunque c'e' un motivo per lasciare spazi nel contare.

Anche se il nostro primo programma potrebbe essere facilmente numerato con incrementi di 1 tra una riga e l'altra, non lascerebbe spazio per aggiungere righe in piu' se avessimo bisogno di farlo.

Per vedere come il computer ordina le righe, aggiungi la seguente riga:

```
25 REM UNA RIGA NEL MEZZO
```

Ora scrivi LIST in modo che il computer listi l'attuale programma, e vedrai la riga 25 comparire nell'esatto posto numerico. Ora esegui ancora il programma e scoprirai che la riga 25 non ha fatto assolutamente differenze.

COME INSERIRE NOTE

Perche' il computer ha deciso di ignorare la riga 25? La parola REM sta per remark, cioe' nota, ed e' usata nei programmi quando si vuole includere qualche informazione per chi legge il listato.

In ogni caso il computer ignora le note. Esse ci sono solo per tua convenienza, e per la convenienza del programmatore o di chiunque altro stia leggendo il listato del programma.

Spesso si inserisce la parola REM all'inizio del programma, come questo:

5 REM IL POEMA DI GIANNI E MARIA

In teoria sembrerebbe inutile scrivere cio' e, di fatto, in questo caso, e' di scarso servizio. Ma, guardando altri programmi piu' complicati, ti accorgerai che senza l'istruzione REM avrai difficolta' nel comprendere, attraverso il listato, quello che supponi debbano fare i programmi.

L'istruzione REM serve a ricordare ai programmatori cio' che deve fare ogni sezione all'interno di un programma. Percio' ti suggerisco fin d'ora ad inserire le note nell'interno dei tuoi programmi.

ANCORA SU PRINT

Torniamo al comando PRINT. Cancella il programma che c'e' in memoria inserendo NEW e premendo poi RETURN. Batti ora il seguente programma nel tuo computer e quindi eseguilo:

```
10 PRINT 1,2
15 PRINT
20 PRINT 1;2;3
25 PRINT
30 PRINT "COMPUTER"
35 PRINT
40 PRINT "23 + 34 =";23 + 34
45 PRINT
50 PRINT 2*3
55 PRINT
60 PRINT 3↑5
65 PRINT
70 PRINT "LA RISPOSTA E' ";23 + 5 - 7/6
```

C'e' molto da imparare in questo programma.

Prima di tutto come nel programma di Gianni e Maria, il computer esegue il programma riga per riga, partendo dal numero piu' piccolo, procedendo ordinatamente attraverso i numeri delle righe fino a che questi finiscono, quando si ferma.

(Scoprirai che questo seguire sequenzialmente il numero di linea non si applica sempre, siccome ci sono modi che permettono al computer di eseguire parti di un programma al di fuori dell'ordine numerico, ma per adesso, e' meglio affermare che il programma sara' eseguito seguendo l'ordine numerico delle linee).

Guarda per prima la riga 10 del tuo programma. Ti accorgerai che c'e' una virgola tra 1 e 2. Questo fa si' che il computer lasci uno spazio tra i numeri. Quando usi la virgola in questo modo ti sembrera' che essa divida lo schermo in piccole file ben allineate. Prova a stampare 1,2,3,... in modo da capire perfettamente cio' che sta avvenendo.

La seconda riga del programma, 15, e' composta soltanto dalla parola PRINT. Come potrai osservare sullo schermo, cio' permette di inserire una riga vuota tra quelle righe che contengono informazioni dopo la parola PRINT. Lo stesso accade per le righe 25, 35,...65.

La riga 20 ha tre numeri (1,2 e 3) separati da un punto e virgola. Invece di separare i numeri, come fa la virgola, vedrai che in questo caso i numeri vengono separati da un singolo spazio. Userai il punto e virgola quando vorrai che informazioni stampate seguano altre informazioni senza altri spazi inutili.

La riga 30 e' una stringa perche' e' racchiusa tra virgolette.

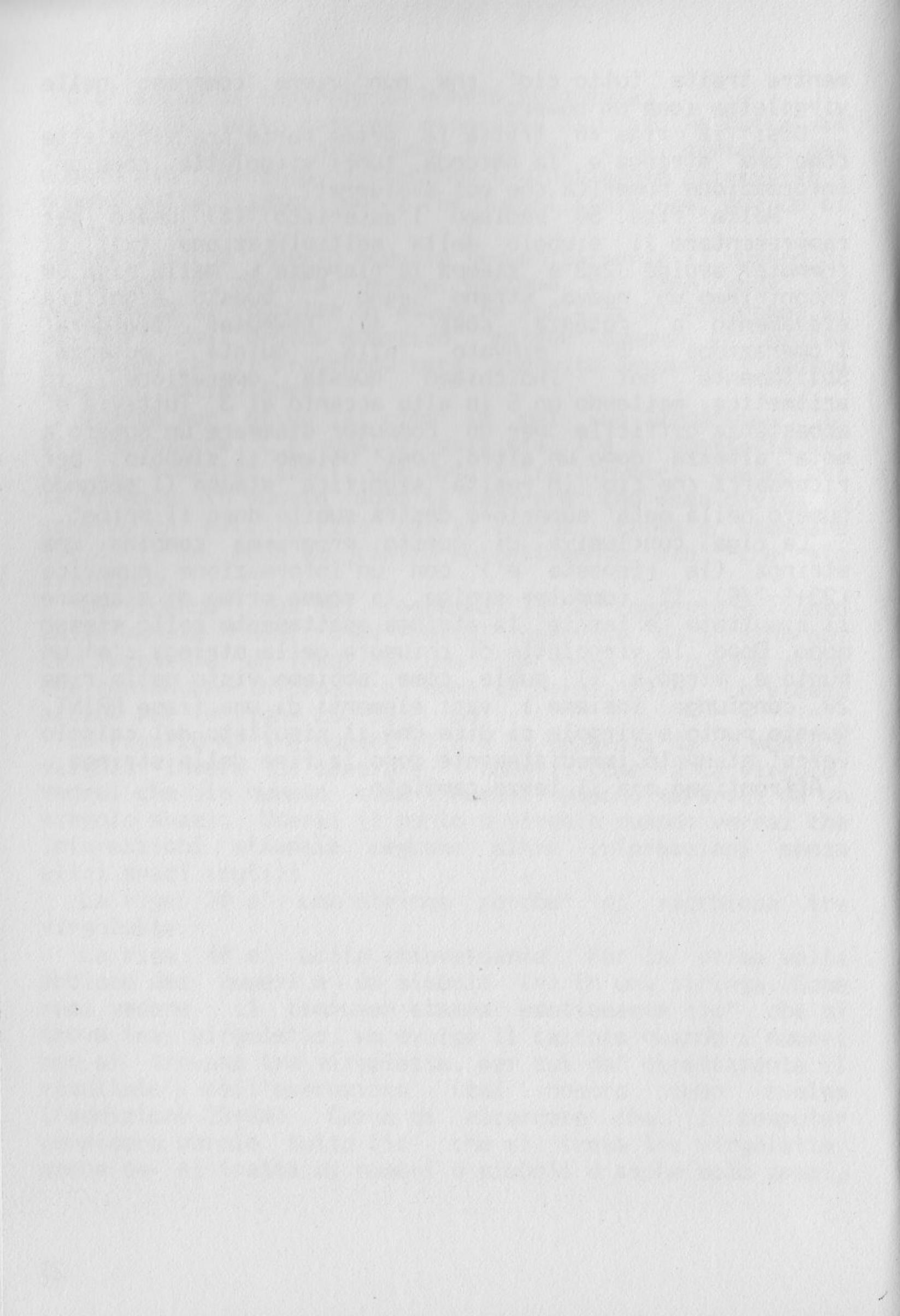
La riga 40 e' molto interessante. Per la prima volta abbiamo dei numeri e un simbolo (=) in una stringa. Come puoi vedere il computer stampa esattamente cio' che si trova tra virgolette, ma svolge il calcolo quando i numeri non si trovano tra virgolette, per cui da' direttamente il risultato dell'operazione (nel nostro caso svolge l'addizione 23+34). Cerca di ricordare che il computer considera parole tutto cio' che si trova tra virgolette, anche se si tratta di numeri o simboli o anche solo spazi,

mentre tratta tutto cio' che non viene compreso nelle virgolette come un numero.

Cosi' la riga 40 tratta la prima parte tra virgolette come una stringa e la seconda, fuori virgolette, come un' informazione numerica che poi svolgera'.

Nella riga 50 vediamo l'asterisco (*) usato per rappresentare il simbolo della moltiplicazione (x); il computer svolge 2x3 e stampa la risposta 6. Nella riga 60 incontriamo un nuovo, strano segno . Questo significa elevamento a potenza, cosi' il computer svolgera' l'operazione 3 elevato alla quinta potenza. Solitamente noi indichiamo questa operazione, in aritmetica, mettendo un 5 in alto accanto al 3. Tuttavia e' abbastanza difficile per un computer stampare un numero a meta' altezza dopo un altro, cosi' usiamo il simbolo per ricordarci che cio' in realta' significa "stampa il secondo numero nella meta' superiore destra subito dopo il primo".

La riga conclusiva di questo programma combina una stringa (la risposta e') con un'informazione numerica (23+5-7/6). Il computer svolge la somma prima di stampare il risultato e lascia la stringa esattamente nello stesso modo. Dopo le virgolette di chiusura della stringa c'e' un punto e virgola, il quale, come abbiamo visto nella riga 20, congiunge insieme i vari elementi di una frase PRINT. Questo punto e virgola ci dice che il risultato del calcolo verra' stampato immediatamente dopo la fine della stringa.

Affrontiamo ora il terzo capitolo.

C A P I T O L O 3

A L T R I   I M P O R T A N T I   C O M A N D I

Ora che hai imparato ad inserire informazioni sullo schermo scoprirai che bisogna essere capaci di cancellare lo schermo per inserire un maggior numero di istruzioni PRINT. Possiamo fare cio' usando il comando per cancellare lo schermo.

CANCELLA LO SCHERMO

```
10 PRINT "PROVA"
20 INPUT A$
30 PRINT "♥"
```

Inserisci il seguente programma nel tuo computer e poi eseguilo (scrivendo RUN):

Il simbolo con il piccolo cuore si ottiene premendo il tasto SHIFT e il tasto CLR/HOME. Quando si fa cio' tra virgolette, appare il simbolo con il cuore che il computer legge come: "cancella lo schermo quando il programma viene fatto girare (comando RUN)". Se si premono i tasti SHIFT e CLR/HOME senza le virgolette, il computer cancellera' immediatamente lo schermo.

Quando tu esegui il programma, vedrai la parola "prova" apparire in cima allo schermo. Sotto comparira' un punto interrogativo (?). Da dove e' venuto? Il punto interrogativo e' conosciuto come un "input prompt". Un input prompt appare in un programma quando il computer aspetta che tu inserisca qualcos'altro nella macchina, o che tu prema semplicemente RETURN. Nella riga 20, il computer sta aspettando (INPUT) una stringa (perche' la A che segue la parola INPUT e' seguita da un segno dollaro - $-). Tu puoi inserire una parola, un numero, una combinazione di parole, numeri o simboli in risposta ad una INPUT stringa. (Ma puoi scrivere solo un numero in risposta ad una INPUT numerico. Se premi solamente il tasto RETURN

quando il computer vuole un numero, esso pensera' che tu voglia lo zero -0-).

Per quanto riguarda il nostro programma, quando tu rispondi all'input prompt, dopo aver premuto RETURN, vedrai lo schermo cancellarsi e la parola "prova" scomparire. Dove e' andata? Abbiamo detto che il computer svolge il programma seguendo l'ordine numerico delle righe. Prima di tutto il programma ha stampato, con la riga 10, "prova" sullo schermo, poi e' andato avanti alla riga 20 dove ha aspettato un input (e tu che premi il tasto RETURN). Una volta che hai fatto cio' nella riga 20, il computer si e' spostato alla riga 30 dove, trovando il simbolo significante "cancella lo schermo", ha eseguito l'operazione.

Prova il programma molte volte fino a che tu non abbia acquisito familiarita' con l'operazione.

CANCELLARE LO SCHERMO AUTOMATICAMENTE

Invece di aspettare che il tasto RETURN venga premuto, puoi scrivere un programma che cancelli lo schermo automaticamente, come dimostra il nostro prossimo esempio. Inserisci questo nuovo programma nel tuo computer, batti RUN e poi RETURN.

```
10 PRINT "AUTOPROVA"
20 FOR A=1 TO 500
30 NEXT A
40 PRINT "□"
50 FOR A=1 TO 500
60 NEXT A
70 GOTO 10
```

Quando il programma gira, vedrai la parola "autotest" lampeggiare in cima allo schermo. Guardiamo il programma ed esaminiamolo riga per riga. Prima di tutto, alla riga 10, il computer stampa la parola "autotest" sullo schermo. Secondariamente, alla riga 20, incontriamo la parola FOR. Ci soffermeremo sulla istruzione FOR/NEXT in seguito, ma tutto cio' che ci basta conoscere ora e' che il computer usa FOR/NEXT per contare. In questo programma le righe 20 e

30 diconо al computer di contare fino a 500 prima di andare avanti.
Per fermare il programma, premi STOP.
In questo modo il computer attende un attimo mentre sta contando da 1 a 500. Poi passa alla riga 40 in cui troviamo l'istruzione PRINT "" che dice al computer di cancellare lo schermo. Il computer poi, nelle righe 50 e 60, incontra ancora FOR/NEXT cosi' aspetta un attimo e ricomincia a contare da 1 a 500. Andando avanti nella sequenza arriva alla riga 70 dove trova l'istruzione GOTO 10. Questo, ovviamente, ordina al computer di ritornare alla riga 10. Quando il computer arriva alla riga 70, obbedisce all'istruzione GOTO, e continua dalla riga 10 passando attraverso la stampa di "autotest", contando fino a 500, cancellando lo schermo,contando ancora fino a 500 e poi arrivando a GOTO 10 e ricominciando tutto di nuovo.

PER CAMBIARE FACILMENTE IL CONTENUTO DELLE RIGHE NEI PROGRAMMI

Nel tuo computer si trova una funzione EDIT che rende molto semplice cambiare i contenuti delle righe nei programmi. Cancella dalla memoria il precedente programma, scrivendo la parola NEW e premendo poi il tasto RETURN. Inserisci poi il seguente programma. NON FAR GIRARE QUESTO PROGRAMMA.

```
10 REM QUESTO E' UN TEST EDOD
20 REM QUESTO E' UN UN TEST EDIT
30 REM QUESTO E' TEST EDIT
```

Se vuoi cambiare le righe presenti sullo schermo, tutto cio' che devi fare e' usare i tasti CURSOR che si trovano nella parte inferiore destra della tastiera, muovere il cursore nella posizione desiderata e fare i dovuti cambiamenti.
Per correggere l'errore di scrittura nella riga 10, usa il tasto SHIFT e il tasto CRSR con le frecce verticali fino

ad arrivare alla riga 10. Ora con il tasto CRSR e le frecce orizzontali, ti muovi verso destra fino a che il cursore non lampeggi sopra la "O" di EDOD. Schiaccia adesso il tasto "I" e poi quello "T". Avrai cosi' corretto la parola, ma per inserire la riga cambiata nella memoria del computer e cancellare la riga errata, devi premere RETURN.

Passiamo ora alla riga 20. Abbiamo la parola "UN" scritta due volte, muoviamo il cursore fino alla "E" di EDIT e premiamo il tasto DEL (che sta per "delete", che significa cancellare) tre volte, due per la parola "UN" e una per lo spazio. Il tasto DEL rimuovera' il carattere alla sinistra del cursore e rimettera' a posto la riga corretta.

Nella riga 30 una parola (UN) e' stata dimenticata. Muovi il cursore fino alla "E" di EDIT e poi premendo il tasto SHIFT, premi anche il tasto INST (che signifa inserisci e si trova sullo stesso tasto di DEL) tre volte. Questo ti aprira' la riga in modo che tu potrai scrivere la parola "UN". Non dimenticare di premere il tasto RETURN per inserire la nuova riga nella memoria del computer.

Per vedere il nostro nuovo programma scrivi LIST e schiaccia RETURN; questo ti mostrera' sullo schermo le nuove righe corrette.

I tasti CURSOR e il tasto INST/DEL hanno una funzione ripetitiva; cosi', se tu tieni premuti questi tasti, essi continueranno a ripetere la loro funzione fino a che non saranno lasciati andare. Puoi naturalmente usare le funzioni EDIT anche mentre sei in fase di scrittura del programma. Non e' necessario aspettare che tutto il programma sia stato battuto per poi tornare indietro e correggere gli errori.

Il tasto HOME riportera' il cursore all'angolo superiore sinistro dello schermo; premendo lo stesso tasto con il tasto SHIFT otterrai l'altra funzione, CLR, che cancellera' tutto lo schermo, ma non tocchera' alcun programma che si trovi gia' nella memoria del COMMODORE 64.

Per correggere una riga che non si trova al momento sullo schermo, scrivi LIST seguito dal numero di riga richiesto. Dopo aver premuto RETURN, questa riga apparira' sullo schermo pronta per essere corretta. Se desideri cancellare una riga intera, scrivi soltanto il numero della riga e poi premi RETURN. Avrai cosi' cancellato la riga dalla memoria.

COME RIVEDERE TUTTO UN PROGRAMMA

Se vuoi vedere un listato completo dopo che si e' cancellato, una volta che avrai fatto girare il programma, tutto cio' che devi fare e' scrivere la parola LIST e poi premere il tasto RETURN. Non c'e' ragione per cui, usando LIST, tu debba avere il listato per forza dall'inizio. Quando hai programmi un po' lunghi, potresti ad esempio volere solo una parte del listato. Puoi fare cio' usando il simbolo (-) come segue:

```
LIST -100     lista fino alla linea 100 inclusa
LIST 50-90    lista dalla riga 50 alla 90
LIST 150-     lista dalla riga 150 alla fine
LIST 270      lista solo la riga 270
```

COME USARE LA STAMPANTE

Moltissime indicazioni dovrebbero essere date per il corretto uso della stampante, ma se preferisci non occupartene in questo momento e controllare soltanto che la tua stampante funzioni, questi sono i comandi di cui hai bisogno:

OPEN fn,dn

Questo comando ti permette di comunicare con una periferica connessa al tuo computer, ad esempio una stampante o un disk drive. OPEN deve essere seguito da un numero di file (fn), che e' un numero compreso tra 1 e 255. Non importa il numero usato, basta che sia un numero presente nel tuo set di comandi. Il numero di file deve essere seguito da una virgola e da un numero di periferica (dn). La tua stampante solitamente ha il numero 4 di periferica e il disk drive ha il numero 8. Cerca sul manuale il tuo corretto numero di periferiche.

CMD fn

Questo comando trasferisce il controllo dal computer alla stampante e deve essere seguito dal numero di file che hai usato con il comando OPEN.

Una volta usati il comando OPEN e CMD, per listare un programma scrivi LIST e premi RETURN proprio come agiresti per far apparire il listato sullo schermo. Infatti dopo aver usato OPEN e CMD puoi inserire tutti i giochi che abbiamo visto e spostarli sulla stampante invece che farli apparire sullo schermo.

PRINT# fn

Questo comando viene usato per stampare informazioni e per chiudere il canale di comando che si era aperto con CMD.

CLOSE fn

Quando hai finito di usare il numero di file inserisci il comando CLOSE, per disinserirlo. Puoi avere soltanto 10 file aperti contemporaneamente.

Per far in modo che il tuo programma scorra sulla stampante invece che sullo schermo, tutto cio' che ti si richiede e' di includere i comandi OPEN e CMD nelle prime righe del programma. Non dimenticare di usare PRINT# e CLOSE per chiudere il canale di comando e il file.

IL DOS

DOS significa Disk Operating System. In questa sezione parleremo della memoria di massa a disco del COMMODORE. Se non hai un disk drive o questa operazione ti e' gia' familiare, passa oltre. Esamineremo la memoria di massa a disco e i comandi di cui tu hai bisogno perche', in poco tempo, ti diventi facile capire il modo in cui dovrai usarlo. Piu' dettagliate istruzioni sara' possibile trovarle nell'opuscolo che accompagna il drive.

Prima di tutto vedi se hai ben connesso il drive con il computer. Quando ti appresti ad accendere, sii sicuro di aver acceso per primo il drive e poi il computer. Quando hai altre periferiche inserite nel computer, questi dovrebbe essere sempre l'ultimo ad essere acceso. E' anche raccomandabile di non lasciare il dischetto nel disk drive quando lo si sta accendendo o spegnendo. Potrebbe accadere

che le informazione contenute nel dischetto vengano alterate. Quando accendi il drive, sia la luce verde che quella rossa si illumineranno; quella rossa, dopo qualche secondo, si spegnera'. Ora accendi il computer, la luce rossa comparira' ancora per rispegnersi nuovamente dopo poco, e il computer sara' pronto per funzionare.

Esamineremo ora i comandi piu' importanti per salvare, caricare e cancellare programmi su disco. Quando devi svolgere queste operazioni su disco, devi dare un nome al programma. Quando usi i tuoi programmi sei tu stesso che decidi quale nome dare al programma per un massimo di 16 caratteri. Se stai usando programmi commerciali, il disco programmato deve giungere gia' con determinate istruzioni che includeranno anche il nome del programma. Devi dire anche al computer quale periferica desideri usare. Ricorda che il numero di periferica per il drive e' 8. Se avrai un secondo drive, il suo numero di periferica sara' 9.

COME CARICARE UN PROGRAMMA

Per caricare un programma nella memoria del computer, scrivi LOAD"nome del programma",8. Il computer ti dara' allora il messaggio SEARCHING FOR nome del programma. Quando lo avra' localizzato, stampera' il messaggio LOADING, e poi READY quando avra' finito. Ora potrai far girare il programma. Ricorda che mentre il drive sta caricando il programma, la luce rossa rimarra' accesa.

COME SALVARE UN PROGRAMMA

Prima di poter salvare un programma su un nuovo disco, il disco deve essere formattato. Cio' significa che il DOS nel drive crea una Block Availability Map (BAM) che tiene una registrazione di quali parti della memoria del disco sono libere e quali sono state gia' occupate da altro materiale. Esso crea anche un indice, o Directory, dei programmi presenti su disco. Questo indice puo' essere caricato nella memoria del computer e listato in modo tale che tu possa vedere quali programmi sono presenti su disco. Per vedere il directory scrivi LOAD"$",8. Il segno del dollaro significa directory. Quando il computer stampa READY,

ricordati di listare ma non di far girare il directory.

Per formattare un nuovo disco, mettilo nel drive e scrivi OPEN 15,8,15,"NEW8:nome del disco,ID". Abbiamo trattato il comando OPEN nella sezione sulla stampante; il primo 15 e' il numero di file; l'8 e' il numero di periferica; il secondo 15 e' il canale di comando. Non dimenticare le virgole. Il comando NEW puo' essere abbreviato in N. Il nome del disco che usi in questo comando, e' poi dato all'intero disco e appare in cima al directory, quando questo viene listato. La ID significa identita', cioe' uno dei due caratteri che il DOS utilizza per identificare il disco. La ID appare accanto al nome del disco in cima al directory ed anche dopo ognuna delle parti memorizzate dal disco.

Quando scrivi i dati su disco il DOS usa la ID per essere sicuro che i dischi non facciano confusione.

N.B.:Il comando NEW cancella completamente il contenuto di un disco, cosi', non formattare un disco che contiene gia' dei programmi a meno che tu non voglia il disco pulito.

Se hai gia' usato il comando OPEN e non l'hai ancora chiuso, qualsiasi altro tentativo di usare OPEN dara' come risultato il messaggio FILE OPEN error. Per ovviare a cio' sostituisci OPEN 15,8,15 con PRINT#15.

Tutto cio' sembra molto complicato, ma tu devi solo usare il comando NEW ogni volta che desideri usare un disco nuovo. Dopo che il disco e' stato formattato, puoi cominciare a salvare i programmi su di esso. Per far questo scrivi SAVE"nome del programma",8. Il computer ti dira' allora SAVING nome del programma, il drive comincera' a frullare e il messaggio READY apparira' sullo schermo.

Per essere sicuro che il programma e' stato salvato correttamente, scrivi VERIFY"nome del programma",8. Il computer ti dira' che sta verificando il programma e poi apparira' sullo schermo OK e READY se il programma e' corretto, oppure VERIFY ERROR, se ha trovato un errore nel programma salvato. Questo comporta la cancellazione del programma perche' il DOS non permette di salvare due programmi che abbiano lo stesso nome. Cosi' bisogna cancellare quello salvato erroneamente e salvarlo ancora.

COME CANCELLARE UN PROGRAMMA

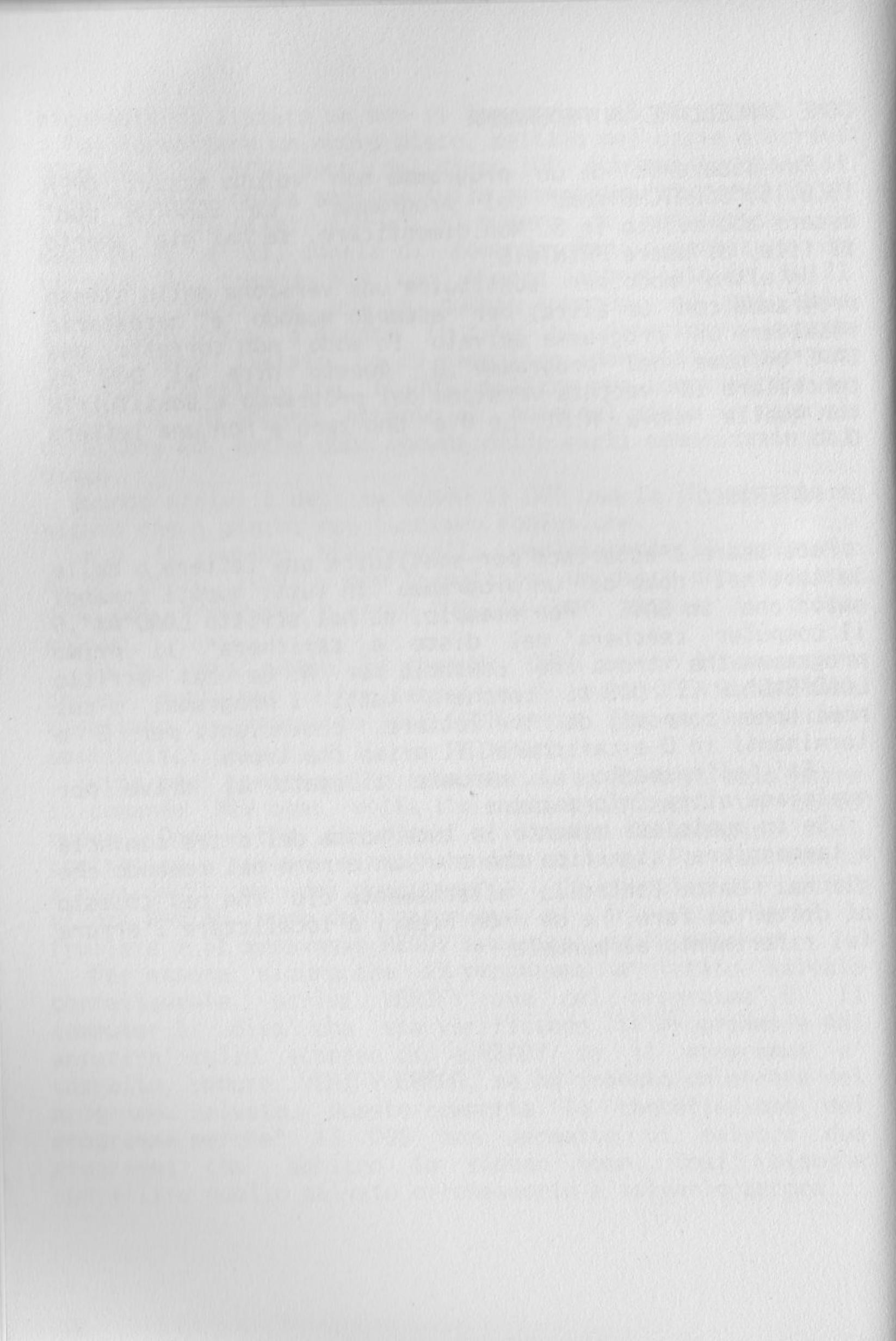
Per liberarci di un programma non voluto scrivi OPEN 15,8,15,"SCRATCH8:nome del programma". Lo SCRATCH puo' essere abbreviato in S. Non dimenticare, se hai gia' aperto il file, di usare PRINT#15.

Un altro modo per sostituire una versione dello stesso programma con un'altra, per esempio quando e' necessario risalvare un programma salvato in modo non corretto, usa SAVE"@0:nome del programma",8. Questo dice al DOS di cancellare la vecchia versione del programma e sostituirla con quella nuova. N.B. Lo 0 e' uno zero e non una lettera O.

L' ASTERISCO

Puoi usare l'asterisco per sostituire una lettera o delle lettere nel nome di un programma in tutti questi comandi salvo che in SAVE. Per esempio, se hai scritto LOAD"A*",8 il computer cerchera' nel disco e carichera' il primo programma che trova che comincia per A. Se hai scritto LOAD"B*D",8 il DOS ti cerchera' tutti i programmi i cui nomi sono composti da tre lettere, comincianti per B e terminanti in D e carichera' il primo che trova.

Fa' riferimento al manuale allegato al drive per qualsiasi altra informazione.

Se in qualsiasi momento la luce rossa del drive comincia a lampeggiare, significa che c'e' un errore nel comando che gli hai dato. Controlla attentamente cio' che hai chiesto al drive di fare, e se non riesci a localizzare l'errore fai riferimento al manuale.

C A P I T O L O   4

I M P A R I A M O   A   P R O G R A M M A R E

E' ora di iniziare a considerare dei veri programmi. Ti sarai accorto, sfogliando il libro, che da ora in poi troverai programmi di una certa lunghezza. Molti conterranno delle parole del linguaggio BASIC che non sono state spiegate. Questo accade perche', siccome i programmi diventano piu' complessi, diventa sempre piu' difficile spiegare tutte le parole di un programma. Comunque questo non e' un problema, perche' probabilmente sarai capace di scoprire il significato di molte parole guardandole nel contesto di una riga o di una sezione del programma.

Noi stiamo cercando di darvi la spiegazione dei piu' importanti comandi disponibili su un computer; ma quando incontri una parola in un programma che ti sembra strana, scrivila ugualmente. Scoprirai che il programma funzionera' perfettamente senza che tu conosca il significato della parola.

LA CASUALITA'

Quando lanci in aria una moneta, sia che il risultato sia testa o che sia croce, dipende dal caso. Lo stesso accade quando vengono lanciati i dadi.

L'abilita' del tuo computer nel generare numeri a caso e' molto utile nella programmazione. La parola in BASIC RND e' un comando che permette di generare numeri a caso.

COME GENERARE NUMERI A CASO

Cominciamo usando RND per creare alcuni numeri a caso. Inserisci il seguente programma, e fallo girare:

```
10 PRINT RND(1);
20 GOTO 10
```

Facendo cio' vedrai una lista di numeri come questa apparire sullo schermo:

```
.845119334  5.16042709E-03  .939567085
 .410618822  .359784936  .387747735  .11
8996772  .026818256  .596047378  .494264
057  .199725193  .463177715  .933745125
 .171695808  .0220181925  .341925621  .5
68434475  .987519491  .220016581  .37116
2869  .747017832  .70300916  .314634204
 .979448431  .698403042  .133029162  .42
2621355  .644702411  .887324984  .964261
51  .957226436  .482780676  .489715975
.89463107  .944857278  .494556765  .6769
30171  .21945822  .0756854787  5.1652704
3E-03  .328395005  .285127397  .12794094
1  .376015229  .431950661  .483941591  .
793402832  .439236734  .969135143
```

Come puoi vedere, RND genera numeri a caso tra 0 e 1. Se lasci girare il programma, esso andra' avanti apparentemente per sempre, scrivendo nuovi numeri a caso sullo schermo.

I numeri a caso tra 0 e 1 hanno un interesse limitato se vogliamo generare numeri che assumano altri significati. Ad esempio se vogliamo usare il computer come una specie di moneta elettronica assegnamo a testa il numero 1 e a croce il numero 2. Se invece vogliamo che il computer generi numeri a caso tra 1 e 6 come se fosse un dado, dobbiamo dargli un comando particolare. Inserisci il seguente programma e fallo girare:

```
10 PRINT INT(6*RND(1))+1;
20 GOTO 10
```

Quando fai girare questo programma, avrai una serie di numeri scelta a caso tra 1 e 6, come questa:

```
  6  3  5  4  3  4  5  1  3  4  2  3  2
 5  1  4  1  3  4  1  6  5  2  4  5  2  4
   3  1  3  3  5  4  5  6  2  5  4  5  2
  6  5  3  1  1  1  3  1  3  5  3  5  4
 5  6  3  1  5  6  3  4  6  2  5  1  1  6
   4  5  3  3  3  2  5  4  1  5  2  1 5
  5  5  5  1  6  5  2  5  3  3  2  6  6
 3  6  6  5  4  4  3  2  4  6  4  1  5  3
   2  2  2  4  3  5  4  1  4  4  4  3  5
  6  5  3  5  3  3  2  6  5  2  6  6  6
 5  2  6  3  6  1  6  1  3  5  4  4  5  3
   6  2  6  2  2  3  1  4  5  5  3  3  2
  1  4  2  2  1  3  1  3  6  2  4  4  5
 1  2  2  6  5  1  4  4  3  6  5  1  2  6
   3  2  4  5  1  3  3  4  3  6  1  2  5
  2  3
```

Guardiamo la riga 10 per vedere come abbiamo ottenuto i numeri tra 1 e 6. Prima di tutto abbiamo moltiplicato il nostro generatore di numeri a caso per 6, per darci i numeri richiesti al posto di numeri decimali minori di 1. Poi abbiamo usato il comando INT. Questo comando dice al computer di darci solo quella parte di numero alla sinistra della virgola e di dimenticare i numeri decimali posti a destra della virgola stessa. Cosi' avremo numeri interi.

Potresti chiedere ora a che cosa serve il +1 alla fine. Sebbene il generatore di numeri a caso dia numeri tra 0 e 1, esso non dara' mai propriamente 0 e 1 come risposta ma tutti i numeri decimali compresi tra di essi. Il piu' piccolo sara' 0,000000001; il piu' grande sara' 0,999999999. Moltiplicando il numero a caso per 6 e considerando solo gli interi, il computer dara' quindi numeri da 0 a 5. Per imitare la gettata di un dado dobbiamo liberarci dello 0 e ottenere un 6. Per fare cio' aggiungiamo un 1 al nostro numero. Puoi usare questo metodo per generare numeri a caso, compresi in qualsiasi intervallo da te desiderato. Per esempio, per ottenere un

numero tra 100 e 150, bisogna moltiplicare il generatore di numeri a caso per 50 e aggiungere 101 al risultato. Non dimenticare di usare INT, per assicurarti di ottenere numeri interi.

Ora possiamo creare una vasta serie di numeri tra 1 e 6 usando il nostro programma per generare numeri a caso ma tutto sommato non e' molto interessante. Esamineremo allora alcuni divertenti modi per impiegare questa funzione.

TAVOLA CALDA

Inserisci e fai girare il seguente programma che utilizza l'abilita' del computer nel generare numeri a caso. Come potrai vedere esso crea una situazione in cui tu lasci a lui la facolta' di decidere quale portata ti farai servire:

```
10 REM TAVOLA CALDA
20 PRINT "♥":REM CANCELLA LO SCHERMO
30 A=INT(4*RND(1))+1
40 PRINT "TU HAI ORDINATO ";
50 IF A=1 THEN PRINT "UN HAMBURGER CON C
IPOLLA"
60 IF A=2 THEN PRINT "UNA FRITTELLA FRAN
CESE"
70 IF A=3 THEN PRINT "FOCACCE FARCITE"
80 IF A=4 THEN PRINT "UNA PIZZETTA"
90 FOR Z=1 TO 1000:NEXT Z
100 PRINT
110 GOTO 30
```

Quando fai girare questo programma, comparira' una lista di portate sullo schermo:

```
TU HAI ORDINATO FOCACCE FARCITE

TU HAI ORDINATO UNA PIZZETTA

TU HAI ORDINATO UN HAMBURGER CON CIPOLLA
```

```
TU HAI ORDINATO UNA FRITTELLA FRANCESE

TU HAI ORDINATO UNA PIZZETTA

TU HAI ORDINATO UNA FRITTELLA FRANCESE

TU HAI ORDINATO FOCACCE FARCITE

TU HAI ORDINATO UNA PIZZETTA

TU HAI ORDINATO UNA PIZZETTA

TU HAI ORDINATO UN HAMBURGER CON CIPOLLA

TU HAI ORDINATO UNA PIZZETTA
```

Se riguardi il listato, ti accorgerai come il programma assegna, nella riga 30, alla lettera A il valore del numero a caso. In questo caso, la lettera A assolve la funzione di un numero. Essa e' chiamata 'variabile numerica'. Nel gergo del computer, si dice che nella riga 30 il computer ha assegnato il valore del numero a caso alla variabile A e come puoi vedere nelle righe seguenti il valore assegnato ad A determina quale portata hai ordinato. Leggi piu' volte il programma se pensi che sia necessario per una miglior comprensione.

Prima di lasciarlo, usiamolo per imparare un nuovo comando. Aggiungi le righe 24, 26 e 27 come da listato:

```
10 REM TAVOLA CALDA
20 PRINT "♥":REM CANCELLA LO SCHERMO
24 INPUT "VUOI ORDINARE ORA? (S/N) ";X$
26 IF X$="N" THEN STOP
27 IF X$<>"S" THEN GOTO 24
30 A=INT(4*RND(1))+1
35 PRINT
40 PRINT "TU HAI ORDINATO ";
50 IF A=1 THEN PRINT "UN HAMBURGER CON C
IPOLLA"
```

```
60 IF A=2 THEN PRINT "UNA FRITTELLA FRAN
CESE"
70 IF A=3 THEN PRINT "FOCACCE FARCITE"
80 IF A=4 THEN PRINT "UNA PIZZETTA"
90 FOR Z=1 TO 1000:NEXT Z
100 PRINT
110 GOTO 30
```

Quando si fa girare la nuova versione del programma qualcosa risulta modificato. La frase "VUOI ORDINARE ORA (S/N)" appare sullo schermo seguita da un punto interrogativo e dal cursore. Questo ci fa vedere che l'INPUT lavora come il comando PRINT quando si stampano informazioni sullo schermo. Ma cos'altro svolge? Il punto di domanda ti segnala che il computer sta aspettando la tua risposta e il cursore indica dove questa apparira' sullo schermo. Il punto e virgola dopo le virgolette di chiusura hanno la stessa funzione svolta nel comando PRINT.

Alla fine della riga troviamo X$. Anche questa e' una variabile in cui il simbolo del dollaro indica che si tratta di una 'variabile stringa'. Ad essa verra' assegnato il valore della tua risposta e nel contempo, servira' a mantenere il valore di lettera invece che di numero. Discuteremo di questo piu' dettagliatamente nei capitoli a seguire.

Dopo aver inserito la risposta e premuto RETURN, la riga 26 esaminera' la risposta per vedere se hai detto "N" che sta per no. Se l'hai fatto il programma si ferma; al contrario, la linea 27 controllera' se hai inserito una risposta non valida ("<>" significa 'diverso da'). Se hai risposta "S", che sta per si' il programma continua generando numeri a caso e offrendoti le portate corrispondenti. Infine e' da notare, in questo programma, la riga 90, usata per inserire una breve pausa. Prova ad eliminare questa riga e ti accorgerai che il programma gira cosi' velocemente che e' quasi impossibile leggere le parole che si susseguono sullo schermo.

C A P I T O L O   5

I L   P R I M O   V E R O   P R O G R A M M A

In questo capitolo introdurremo una parte veramente utile del nostro vocabolario per programmatori: il ciclo FOR/NEXT. Ti ricorderai che abbiamo nominato il ciclo FOR/NEXT quando ti abbiamo insegnato a cancellere lo schermo. E' stato anche usato nel programma 'tavola calda' per immettere una pausa.

Il ciclo FOR/NEXT e' abbastanza semplice. E' formato da due righe nel programma, la prima delle quali e':

```
10 FOR A=1 TO 20
```

e la seconda:

```
20 NEXT A
```

La variabile di controllo, la lettera dopo FOR e NEXT, deve essere la stessa. (Di fatto potresti anche tralasciare la seconda A, siccome il computer sa cosa vuoi dire. Comunque, tralasciando la variabile di controllo, i programmi diventano piu' difficili da leggere, per cui e' raccomandabile scriverla sempre).

Quando si fa girare un ciclo FOR/NEXT, il computer conta dal primo numero fino al secondo come mostrano questi due esempi:

```
10 FOR A=1 TO 20
20 PRINT A;
30 NEXT A
```

Quando lo fai girare vedrai apparire sullo schermo i numeri da 1 a 20.

Ora prova quest'altra versione:

```
10 FOR A=765 TO 781
20 PRINT A;
30 NEXT A
```

Questo e' il risultato:

```
765  766  767  768  769  770  771  772
773  774  775  776  777  778  779  780
781
```

IL PASSO (STEP)

Nei due precedenti esempi, il computer ha contato di unita' in unita', ma non c'e' ragione perche' debba sempre comportarsi in questo modo. La parola STEP puo' essere usata dopo il comando FOR della prima riga, come segue:

```
10 FOR A=0 TO 100 STEP 10
20 PRINT A;
30 NEXT A
```

Quando fai girare questo programma scoprirai che il computer ha contato di decina in decina, producendo questo risultato:

```
0  10  20  30  40  50  60  70  80  90
100
```

Lo STEP non deve necessariamente essere positivo, puo' assumere anche un valore negativo:

```
10 FOR A=100 TO 10 STEP -10
20 PRINT A;
30 NEXT A
```

Ecco cio' che risulta:

```
100  90  80  70  60  50  40  30  20  10
```

COME NIDIFICARE PIU' CICLI

E' possibile inserire piu' cicli FOR/NEXT uno dentro l'altro. Questo procedimento e' chiamato nidificazione di cicli. Nel prossimo esempio, il ciclo B e' inserito nel ciclo A:

```
10 FOR A=1 TO 3
20 FOR B=1 TO 2
30 PRINT A;"VOLTE";B;"E' UGUALE A ";A*B
40 NEXT B
50 NEXT A
```

Il risultato sara':

```
1 VOLTE 1 E' UGUALE A  1
1 VOLTE 2 E' UGUALE A  2
2 VOLTE 1 E' UGUALE A  2
2 VOLTE 2 E' UGUALE A  4
3 VOLTE 1 E' UGUALE A  3
3 VOLTE 2 E' UGUALE A  6
```

Devi stare molto attento ed assicurarti che il primo ciclo che viene aperto deve anche essere l'ultimo ad essere chiuso. Cioe' se FOR A... e' stato il primo ciclo nominato nel programma, l'ultimo NEXT deve essere NEXT A.

Cerca ora di scambiare la riga 10 con la riga 20 nel programma, osserva cosa succede quando mischi i FOR e i NEXT.

Dovresti ricordare che ti ho fatto notare che puoi omettere la variabile di controllo dopo il NEXT. Ti ho anche detto che cio', pero', rende i programmi poco leggibili. Comunque, come immagino tu abbia gia' realizzato, tralasciando le variabili di controllo raggiri il problema di specificare erroneamente il NEXT nei cicli nidificati.

Puoi sostituire le linee 40 e 50 del programma con le seguenti:

```
40 NEXT A:NEXT B
```

```
40 NEXT:NEXT
```

```
40 NEXT A,B
```

TAVOLE MOLTIPLICATIVE

Puoi usare i cicli nidificati per fare in modo che il computer stampi le tavole moltiplicative da 1 per 1 a 12 per 12, come in questo esempio (nota che puoi omettere il punto e virgola tra le parti dell'istruzione PRINT comprese nelle virgolette e quelle al di fuori; questo rende piu' veloce l'esecuzione del programma ma diminuisce la leggibilita' dello stesso):

```
10 FOR A=1 TO 12
20 FOR B=1 TO 12
30 PRINT A;"VOLTE";B;"E' UGUALE A ";A*B
40 NEXT B
50 NEXT A
```

Questa e' una parte del risultato:

```
 5 VOLTE 3 E' UGUALE A  15
 5 VOLTE 4 E' UGUALE A  20
 5 VOLTE 5 E' UGUALE A  25
 5 VOLTE 6 E' UGUALE A  30
 5 VOLTE 7 E' UGUALE A  35
 5 VOLTE 8 E' UGUALE A  40
 5 VOLTE 9 E' UGUALE A  45
 5 VOLTE 10 E' UGUALE A  50
 5 VOLTE 11 E' UGUALE A  55
 5 VOLTE 12 E' UGUALE A  60
 6 VOLTE 1 E' UGUALE A  6
 6 VOLTE 2 E' UGUALE A  12
 6 VOLTE 3 E' UGUALE A  18
 6 VOLTE 4 E' UGUALE A  24
 6 VOLTE 5 E' UGUALE A  30
 6 VOLTE 6 E' UGUALE A  36
 6 VOLTE 7 E' UGUALE A  42
 6 VOLTE 8 E' UGUALE A  48
 6 VOLTE 9 E' UGUALE A  54
```

```
6 VOLTE 10 E' UGUALE A  60
6 VOLTE 11 E' UGUALE A  66
6 VOLTE 12 E' UGUALE A  72
```

Non c'e' ragione perche' entrambi i cicli debbano avere un passo di segno concorde, come dimostra questa variazione nel programma:

```
10 FOR A=1 TO 12
20 FOR B=12 TO 1 STEP -1
30 PRINT A;"VOLTE";B;"E' UGUALE A ";A*B
40 NEXT B
50 NEXT A
```

Questa e' una parte del risultato:

```
3 VOLTE 2 E' UGUALE A  6
3 VOLTE 1 E' UGUALE A  3
4 VOLTE 12 E' UGUALE A  48
4 VOLTE 11 E' UGUALE A  44
4 VOLTE 10 E' UGUALE A  40
4 VOLTE 9 E' UGUALE A  36
4 VOLTE 8 E' UGUALE A  32
4 VOLTE 7 E' UGUALE A  28
4 VOLTE 6 E' UGUALE A  24
4 VOLTE 5 E' UGUALE A  20
4 VOLTE 4 E' UGUALE A  16
4 VOLTE 3 E' UGUALE A  12
4 VOLTE 2 E' UGUALE A  8
4 VOLTE 1 E' UGUALE A  4
5 VOLTE 12 E' UGUALE A  60
5 VOLTE 11 E' UGUALE A  55
5 VOLTE 10 E' UGUALE A  50
5 VOLTE 9 E' UGUALE A  45
5 VOLTE 8 E' UGUALE A  40
5 VOLTE 7 E' UGUALE A  35
5 VOLTE 6 E' UGUALE A  30
```

INDOVINA IL CODICE

Ecco il nostro primo vero programma. In questo gioco,

chiamato CODEBREAKER, che usa svariati cicli FOR/NEXT, il computer pensa un numero di quattro cifre (come 5462) e tu hai otto possibilita' per indovinarlo. In CODEBREAKER, basato su un programma di Adam Bennett e Tim Summers, non solo devi scoprire i quattro numeri che il computer ha scelto, ma anche determinare l'ordine in cui essi si trovano.

Dopo ogni tentativo, il computer ti dira' quanto sei andato vicino alla soluzione finale. Un quadratino bianco corrisponde alla cifra giusta nella posizione sbagliata. Quello nero, al contrario, ti indichera' la cifra giusta nella giusta posizione. Ovviamente quattro quadratini neri testimonieranno la scoperta del codice. Le cifre possono essere ripetute.

Inserisci il programma e prova a giocare contro il computer. Poi ritorna al libro e leggi la spiegazione del programma che meglio chiarira' il ruolo giocato dai cicli FOR/NEXT:

```
10 PRINT "⌐"
20 PRINT:PRINT:PRINT:PRINT "π":REM GIALL
O
30 FOR R=1 TO 40:PRINT CHR$(113);:NEXT
35 PRINT
40 PRINT TAB(14) "CODEBREAKER"
45 PRINT
50 FOR R=1 TO 40:PRINT CHR$(113);:NEXT
60 PRINT
70 PRINT TAB(5) "QUANDO TI SI DICE DI FA
R COSI'"
80 PRINT TAB(5) "INSERISCI UN NUMERO DI
4 CIFRE"
90 PRINT TAB(5) "E POI PREMI RETURN"
100 PRINT
110 PRINT TAB(5) "LE CIFRE POSSONO ESSER
E RIPETUTE"
120 PRINT
130 PRINT TAB(5) "HAI 8 POSSIBILITA' PER
 SCOPRIRE"
140 PRINT TAB(5) "IL CODICE DEL COMPUTER
"
```

```
145 PRINT
150 FOR R=1 TO 40:PRINT CHR$(113);:NEXT
160 FOR Z=1 TO 4000:NEXT
170 PRINT "♫"
180 DIM B(4)
190 DIM D(4)
200 H=0
210 FOR A=1 TO 4
220 B(A)=INT(9*RND(1))
230 NEXT A
240 FOR C=1 TO 8
250 PRINT
260 PRINT TAB(2) "▥ INSERISCI IL TUO TEN
TATIVO NUMERO";C:REM GIALLO
270 INPUT X
275 PRINT:PRINT
280 IF X>9999 THEN GOTO 260
290 IF X<1000 THEN GOTO 260
300 P=INT(X/1000)
310 Q=INT((X-1000*P)/100)
320 R=INT((X-1000*P-100*Q)/10)
330 S=INT(X-1000*P-100*Q-10*R)
340 D(1)=P
350 D(2)=Q
360 D(3)=R
370 D(4)=S
380 FOR E=1 TO 4
390 IF D(E)<>B(E) THEN GOTO 440
400 PRINTTAB(8)"▨■ ■ ";:REM STAMPA IN RE
VON LO SPAZIO NERO E IN REVOFF LO SPAZIO

410 B(E)=B(E)+10
420 D(E)=D(E)+20
430 H=H+1
440 NEXT E
450 IF H=4 THEN FOR D=1 TO 1000:NEXT:GOT
O 650
460 FOR F=1 TO 4
470 D=D(F)
480 FOR G=1 TO 4
490 IF D<>B(G) THEN GOTO 530
```

```
490 IF D<>B(G) THEN GOTO 530
500 PRINTTAB(8) "▚▀ █";:REM STAMPA IN RE
VON LO SPAZIO BIANCO IN REVOFF LO SPAZIO

510 B(G)=B(G)+10
520 GOTO 540
530 NEXT G
540 NEXT F
550 FOR G=1 TO 4
560 IF B(G)<10 THEN GOTO 580
570 B(G)=B(G)-10
580 NEXT G
590 H=0
600 PRINT
610 NEXT C
620 PRINT:PRINT TAB(8) "▥ NON HAI INDOVI
NATO VERO?":REM GIALLO
630 PRINT:PRINT TAB(8) "LA RISPOSTA E':"
;B(1);B(2);B(3);B(4)
640 END
650 PRINT "▣":PRINT:PRINT:PRINT TAB(8) "
▥ BEN FATTO CODEBREAKER !!":REM GIALLO
660 PRINT
670 PRINT:PRINT TAB(3) "▥ HAI INDOVINATO
 IN ";C;"TENTATIVI":REM GIALLO
690 PRINT
700 PRINT
710 END
```

Ecco come appare lo schermo all'inizio del programma:

```
●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●

              CODEBREAKER

●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●

     QUANDO TI SI DICE DI FAR COSI'
     INSERISCI UN NUMERO DI 4 CIFRE
     E POI PREMI RETURN
```

```
LE CIFRE POSSONO ESSERE RIPETUTE

HAI 8 POSSIBILITA' PER SCOPRIRE
IL CODICE DEL COMPUTER

●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●
```

Ed ecco la fine di un tentativo:

```
INSERISCI IL TUO TENTATIVO NUMERO 7
4323

■ ■

INSERISCI IL TUO TENTATIVO NUMERO 8
4625

■ ■ ■

NON HAI INDOVINATO VERO?

LA RISPOSTA E': 4 7 2 5
```

Ora esamineremo il programma, linea per linea, una pratica che seguiremo per molti dei programmi di questo libro. Se non desideri leggere le spiegazioni dettagliate ora (infatti ci sono delle parti che sono abbastanza difficili da capire per il tuo livello), passa oltre e ritornaci piu' tardi quando ne saprai di piu'.

Le righe 30, 50 e 150 stampano un numero di cerchietti per racchiudere il titolo e le istruzioni, insieme alle righe vuote stampate da 35, 45, 60, 100, 120. La riga 160 ferma per alcuni secondi il titolo e le istruzioni sullo schermo, in modo tale da poterle leggere prima che lo schermo sia cancellato dalla riga 170. Gli arrays sono dimensionati nelle righe 180, 190. Parleremo degli arrays in un capitolo a seguire. Per adesso, tutto cio' che e'

necessario sapere e' che, scrivendo DIM B(4), dici al computer che vuoi creare una lista di dati chiamata B, in cui il primo si chiama B(1), il secondo B(2) e cosi' via. Non ti e' necessario dimensionare un array quando hai meno di 11 elementi, ma ti aiuta a ricordare di dimensionarli sempre prima di usarli in un programma. In questo programma gli arrays sono usati per immagazzinare i numeri scelti dal computer e per immagazzinare le cifre che ogni volta scegli per cercare di indovinare il codice.

H e' una variabile numerica che e' inizialmente posta uguale a zero nella linea 200. Nella linea430, si aggiunge 1 al valore di H ogni volta che una cifra viene scoperta al posto giusto, cosi' che, se il valore di H raggiunge 4, il computer viene a sapere che tutte le cifre sono state indovinate e va allora alla riga 650 per stampare le congratulazioni.

Le righe da 210 a 230 generano le quattro cifre che tu dovrai indovinare. La riga 220 usa la funzione RND che abbiamo visto prima, per avere quattro numeri a caso tra zero e nove; ciascuno di loro viene anche immagazzinato negli elementi dell'array B. Nota che il primo ciclo FOR/NEXT del nostro programma compare qui. La A nella riga 210 vale uno, la prima volta che il ciclo viene eseguito, due, la seconda volta e cosi' via in modo tale che cambi anche A nella riga 220.

Il nostro prossimo ciclo FOR/NEXT, che usa C, comincia nella riga successiva. Esso conta da uno a otto per darti gli otto tentativi. La riga 270 riceve il tuo tentativo dopo che le due righe precedenti ti hanno invitato ad inserirlo. Il tuo tentativo risiede nella variabile numerica X e le righe 280 e 290 si assicurano che tu non abbia inserito un numero maggiore o minore di quattro cifre. Se cio' accade il programma torna indietro alla riga 260, per chiederti un'altra volta di inserire un tentativo. La sezione seguente del programma, fino alla riga 590, analizza le tue quattro cifre per dirti il risultato da te ottenuto, usando alcuni cicli FOR/NEXT (380-440, 460-540, 480-530 e 550-580). La riga 610 manda il programma indietro per continuare il ciclo FOR/NEXT riferito a C; se questo e' gia' stato completato, il programma non va alla 250 ma va dalla riga 610 alla 620 per comunicarti che non hai

indovinato il codice negli otto tentativi a tua disposizione. Le righe 630 e 635 stampano sullo schermo il codice generato dal computer. Se invece lo hai indovinato, cosi' che H equivalga a quattro, nella riga 450, allora il programma salta alla riga 650 per stampare il messaggio di congratulazioni.

Nelle linee 25, 260, 400, 500, 620, 655 e 670 abbiamo usato i simboli che permettono di cambiare il colore delle istruzioni PRINT quando queste appaiono sullo schermo. Per fare in modo che il programma sia piu' facile da scrivere, abbiamo incluso delle istruzioni REM alla fine di ognuna di queste righe per spiegare il significato dei simboli. Quelli delle righe 10 e 170 permettono di cancellare lo schermo.

C A P I T O L O   6

I S A L T I P O S S I B I L I N E I P R O G R A M M I

Abbiamo affermato all'inizio di questo libro che, in parecchie situazioni, il computer esegue un programma in ordine di righe, partendo dal numero piu' basso e seguendo l'ordine fino a che il programma raggiunge la riga finale.

Questo non e' sempre vero. Il comando GOTO fa in modo che il programma continui ad essere eseguito dalla riga da te indicata. Inserisci il seguente programma e prima di farlo girare, vedi se riesci a predire quale sara' il risultato dell'esecuzione:

```
10 GOTO 40
20 PRINT "QUESTA E' LA LINEA 20"
30 GOTO 60
40 PRINT "QUESTA E' LA LINEA 40"
50 GOTO 20
60 PRINT "QUESTA E' LA LINEA 60"
70 FOR Z=1 TO 1000:NEXT Z
80 GOTO 40
```

Questo programma un po' pazzo fa saltare il povero computer da una parte all'altra, cambiando posizione nel programma ogni volta che arriva a un'istruzione GOTO. Ecco cio' che dovresti vedere sullo schermo:

```
QUESTA E' LA LINEA 40
QUESTA E' LA LINEA 20
QUESTA E' LA LINEA 60
QUESTA E' LA LINEA 40
QUESTA E' LA LINEA 20
QUESTA E' LA LINEA 60
QUESTA E' LA LINEA 40
```

```
QUESTA E' LA LINEA 20
QUESTA E' LA LINEA 60
QUESTA E' LA LINEA 40
QUESTA E' LA LINEA 20
QUESTA E' LA LINEA 60
QUESTA E' LA LINEA 40
QUESTA E' LA LINEA 20
QUESTA E' LA LINEA 60
QUESTA E' LA LINEA 40
QUESTA E' LA LINEA 20
QUESTA E' LA LINEA 60
```

Il programma comincia alla riga 10 e, trovandovi un GOTO 40, salta alla linea 40 per stampare il messaggio "questa e' la linea 40". Poi continua dalla riga 50 dove trova l'istruzione GOTO 20. Qui obbedisce al comando di stampare "questa e' la linea 20" e poi va alla linea 30 e avanti cosi' fino a che arriva alla riga 70 dove trova il ciclo FOR Z che inserisce una breve pausa prima che il computer vada alla linea 80, per trovare ancora un'altra istruzione GOTO 40 da dove si riprende di nuovo il tutto.

L'ISTRUZIONE DI CONTROLLO IF...THEN

In gergo, questo modo di usare GOTO e' chiamato salto incondizionato; non essendo condizionato il comando, il computer gli obbedisce sempre. Al contrario, una coppia di parole, IF e THEN, quasi sempre impiegate insieme, impongono condizioni all'istruzione GOTO. Cio' vuol dire che: IF qualcosa e' vero, THEN fai qualcos'altro; IF sei affamato, THEN ordina un hamburger.

Il prossimo programma, che simula la gettata di un dado (usando il generatore di numeri a caso), stampa i risultati delle gettate considerandole come parole, ed usa piu' volte l'istruzione IF...THEN:

```
10 REM IL GIOCO DEI DADI
20 GOTO 140
30 PRINT "UNO"
```

```
40 GOTO 140
50 PRINT "DUE"
60 GOTO 140
70 PRINT "TRE"
80 GOTO 140
90 PRINT "QUATTRO"
100 GOTO 140
110 PRINT "CINQUE"
120 GOTO 140
130 PRINT "SEI"
140 A=INT(6*RND(1))+1
150 FOR Z=1 TO 500:NEXT Z
160 IF A=1 THEN GOTO 30
170 IF A=2 THEN GOTO 50
180 IF A=3 THEN GOTO 70
190 IF A=4 THEN GOTO 90
200 IF A=5 THEN GOTO 110
210 IF A=6 THEN GOTO 130
```

Ecco cio' che apparira' sullo schermo quando farai girare il programma:

```
CINQUE      UNO      CINQUE
QUATTRO     DUE      UNO
CINQUE      SEI      SEI
DUE         SEI      QUATTRO
QUATTRO     SEI      TRE
TRE         UNO      DUE
TRE         SEI      QUATTRO
```

SUBROUTINES, UN ALTRO MODO DI SALTARE

C'e' un altro modo per indirizzare il computer durante il corso di un programma. Questo accade usando le 'subroutines' (sottoprogrammi). Una subroutine e' una parte di programma che puo' essere eseguita piu' volte nel corso di esso ed e' piu' efficientemente tenuta al di fuori del programma principale piuttosto che all'interno di esso.

Il prossimo programma lo rendera' chiaro. In esso, il

computer getta un dado piu' volte. La prima volta, il computer lo getta per te; la seconda volta, getta il dado per se stesso. Dopo che ogni coppia di dadi e' stata gettata esso annuncera' il vincitore (ovviamente vince chi ottiene il numero piu' alto). Il programma usa una subroutine per gettare il dado, cosi' che non sono necessarie due identiche routine per la gettata dei dadi all'interno di un singolo programma. Inserisci e fai girare il programma, poi ritorna al libro e ti sara' spiegato dove si trova la subroutine nel programma e come lavora:

```
10 FOR Z=1 TO 1000:NEXT Z
20 PRINT:PRINT
30 FOR C=1 TO 2
40 GOSUB 130
50 IF C=1 THEN A=D
60 IF C=2 THEN B=D
70 NEXT C
80 IF A>B THEN PRINT "HO VINTO"
90 IF A<B THEN PRINT "HAI VINTO"
100 IF A=B THEN PRINT "QUESTO E' UN PARE
GGIO"
120 GOTO 10
130 REM THIS IS A SUB ROUTINE
140 D=INT(6*RND(1))+1
150 IF C=1 THEN PRINT "HO FATTO ";D
160 IF C=2 THEN PRINT "TU HAI FATTO";D
170 FOR Z=1 TO 500:NEXT Z
180 RETURN
```

Ecco cio' che appare quando si fa girare il programma:

```
HO FATTO  2
TU HAI FATTO 2
QUESTO E' UN PAREGGIO


HO FATTO  4
TU HAI FATTO 5
HAI VINTO
```

```
HO FATTO  4
TU HAI FATTO 5
HAI VINTO


HO FATTO  3
TU HAI FATTO 2
HØ VINTO
```

Il programma si ferma per un momento in riga 10, stampa due righe vuote, poi inserisce il ciclo FOR/NEXT C. Quando arriva alla riga 40, che e' percorsa ogni volta che si passa attraverso il ciclo C, il programma e' mandato alla sobroutine partente dalla riga 140. La gettata del dado, simulata nella riga 140, da' il valore alla variabile numerica D. Le due righe seguenti stampano il risultato della gettata, usando un IF/THEN per determinare se la scritta del computer sara' "IO HO FATTO" oppure "TU HAI FATTO" . C'e' una breve pausa e poi il computer va alla parola RETURN. La parola RETURN segnala al computer di ritornare alla linea immediatamente dopo quella che lo ha mandato alla subroutine. In questo programma, questa linea corrisponde alla 50.

Le IF e THEN nelle righe 50 e 60 determinano se il valore della gettata (D) deve essere assegnato alla variabile A o B.

La linea 70 chiude il ciclo FOR/NEXT, poi le righe 80 e 100 determinano se ha vinto il computer (cio' succede se A e' maggiore di B) o se hai vinto tu (se A e' minore di B). Da qui il programma torna indietro alla riga 10 dalla quale ricomincia (in ogni modo, puoi fermare il computer premendo il tasto RUN/STOP). Rileggi questo programma fino a che tu ti senta sicuro di aver compreso il funzionamento delle subroutines).

GETTIAMO ANCORA

Puoi anche modificare un programma in modo che cambi il numero gettato dal dado in parola, usando le subroutines. Malgrado cio' il programma con le subroutine sembra a prima

vista, non molto piu' corto della versione in GOTO, e certamente non e' piu' chiaro. Ecco un modo in cui puo' essere svolto:

```
10 REM IL GIOCO DEI DADI
20 GOTO 140
30 PRINT "UNO"
40 RETURN
50 PRINT "DUE"
60 RETURN
70 PRINT "TRE"
80 RETURN
90 PRINT "QUATTRO"
100 RETURN
110 PRINT "CINQUE"
120 RETURN
130 PRINT "SEI"
140 A=INT(6*RND(1))+1
150 IF A=1 THEN GOSUB 30
160 IF A=2 THEN GOSUB 50
170 IF A=3 THEN GOSUB 70
180 IF A=4 THEN GOSUB 90
190 IF A=5 THEN GOSUB 110
200 IF A=6 THEN GOSUB 130
210 FOR Z=1 TO 500:NEXT Z
220 GOTO 140
```

ON...GOSUB

Comunque c'e' un modo per farlo piu' chiaramente, usando ON...GOSUB. Questo significa che il computer puo' scegliere il numero di riga della subroutine fra un gruppo di numeri dipendenti dal valore che e' stato assegnato alla variabile.

Ecco un'altra versione del programma del gioco dei dadi, usando ON...GOSUB:

```
5 REM IL GIOCO DEI DADI
10 REM ON...GOSUB
20 FOR Z=1 TO 1000:NEXT Z
```

```
30 A=INT(6*RND(1))+1
40 ON A GOSUB 60,80,100,120,140,160
50 GOTO 20
60 PRINT "UNO"
70 RETURN
80 PRINT "DUE"
90 RETURN
100 PRINT "TRE"
110 RETURN
120 PRINT "QUATTRO"
130 RETURN
140 PRINT "CINQUE"
150 RETURN
160 PRINT "SEI"
170 RETURN
```

Guarda prima di tutto la riga 30. Questa assegna un valore, scelto a caso tra uno e sei, alla variabile A. La riga 40 e' la riga piu' importante del programma. Cioe' se A e' uguale a uno, il programma andra' al numero di riga corrispondente al primo numero dopo il comando GOSUB. Se A e' uguale a due, andra' al secondo numero, e cosi' via.

Il programma puo' anche essere abbreviato mediante l'uso dei due punti. I due punti ti permettono di inserire piu' istruzioni su una stessa riga. Quando i RETURN sono posti sulla stessa riga dell'istruzione PRINT, come nel seguente caso, il programma si abbrevia ancora di piu'.

```
5 REM IL GIOCO DEI DADI
10 REM ON...GOSUB
20 FOR Z=1 TO 1000:NEXT Z
30 A=INT(6*RND(1))+1
40 ON A GOSUB 60,70,80,90,100,110
50 GOTO 20
60 PRINT "UNO":RETURN
70 PRINT "DUE":RETURN
80 PRINT "TRE":RETURN
90 PRINT "QUATTRO":RETURN
100 PRINT "CINQUE":RETURN
110 PRINT "SEI":RETURN
```

C A P I T O L O   7

E L A B O R A Z I O N E   D I   U N   P R O G R A M M A

Il COMMODORE 64 ha molte caratteristiche. In questo capitolo vedremo come elaborare un semplice programma ed arricchirlo.

Il programma che abbiamo intenzione di usare e' un gioco facile e veloce che il COMMODORE 64 svolge molto bene. Il seguente listato presenta il gioco dei fiammiferi. Il computer pone sullo schermo un numero a caso di fiammiferi e poi, a turno, entrambi (tu e il computer), dovrete portar via un numero di fiammiferi fino a che non ne rimangano piu' sullo schermo. Colui che prendera' l'ultimo fiammifero, perdera' la partita. Per rendere il gioco un po' piu' difficile, il computer pone un limite al numero di fiammiferi che si possono prendere ad ogni turno.

Il computer gioca molto bene, e, di fatto, non fa mai errori. Come esercizio aggiungi delle informazioni che rendano possibile un errore da parte del computer.

All'inizio del gioco, la linea 40, sceglie il numero di fiammiferi che saranno usati, e la linea 50 si assicura che si tratti di un numero dispari, una richiesta essenziale del gioco. Le righe da 170 a 200 stampano i fiammiferi sullo schermo usando un ciclo FOR/NEXT e CHR$(110), che e' il codice per il carattere grafico sulla facciata frontale destra del tasto N. La riga 190 stampa una riga vuota, perche' la funzione RND genera un numero maggiore di 0,6. Questo ha l'effetto di iniziare una nuova riga di fiammiferi. Ecco perche' la composizione delle righe di fiammiferi varia sullo schermo dopo ogni turno. La linea 290 svolge la mossa del computer.

Il programma cancella lo schermo dopo ogni turno ed usa delle righe vuote per spaziare le righe di fiammiferi sullo schermo.

Scrivi questo listato e giocalo per piu' volte e poi esamineremo il modo per arricchirlo:

```
10 REM FIAMMIFERI
30 E=0
40 Z=16+INT(8*RND(1))
50 IF 2*INT(Z/2)=Z THEN 40
60 H=3+INT(2*RND(1))
70 PRINT "♥":REM CANCELLA LO SCHERMO
80 PRINT
90 PRINT
110 PRINT "IL NUMERO MASSIMO DI FIAMMIFE
RI CHE PUOI PRENDERE E' ";H

120 PRINT
130 IF E>0 THEN PRINT "HAI PRESO ";E;"FI
AMMIFERI"

140 PRINT
150 IF E>0 THEN PRINT "HO PRESO ";Q;"FIA
MMIFERI"

160 PRINT
170 FOR K=1 TO Z
180 PRINT CHR$(110);" ";
190 IF RND(1)>.6 THEN PRINT
200 NEXT K
210 PRINT:PRINT

220 PRINT "QUANTI NE VUOI PRENDERE?"
230 INPUT E
240 IF E>H OR E<1 THEN 230
250 Z=Z-E
260 PRINT
270 IFZ<1 THEN PRINT "HAI PRESO L'ULTIMO
 FIAMMIFERO, HO VINTO! ":END

290 Q=Z-1-INT((Z-1)/(H+1))*(H+1)
300 IF Q<1 OR Q>H THEN 290
310 PRINT
320 Z=Z-Q
330 IF Z=0 THEN PRINT "HO PRESO ";Q;"FIA
MMIFERI"

340 IF Z=0 THEN PRINT "HAI VINTO":END
350 GOTO 70
```

In questa prima versione del programma tutte le informazioni PRINT vengono stampate a partire dal lato sinistro dello schermo. Il primo cambio che opereremo sara' quello di posizionarle, centrandole sullo schermo. Abbiamo visto nei capitoli precedenti che, schiacciando il tasto SHIFT, e il tasto CLR, lo schermo viene cancellato; abbiamo anche visto che inserendo cio' tra virgolette appariva un simbolo che il computer traduceva con "cancella lo schermo" quando faceva girare il programma. Se hai gia' scritto i precedenti programmi del libro ti sarai accorto che ci sono altri simboli per controllare il colore ed anche alcuni movimenti del cursore. Useremo ora nel nostro gioco il comando "cursore destro".

Esamina attentamente questa nuova versione del programma e opera i cambi necessari. Per cambiare una riga muovi il cursore all'inizio della riga poi fallo scorrere fino al carattere a destra del posto in cui tu desideri inserire i nuovi caratteri. Premi INST (usando SHIFT) il numero di volte richiesto e poi aggiungi i nuovi caratteri. Non dimenticare di premere RETURN prima di spostarti sulla prossima riga.

CURSORE SU = ▯

CURSORE GIU' = ▮

CURSORE A SINISTRA = ▮

CURSORE A DESTRA = ▮

CURSORE HOME = ▮

INSERZIONE = ▮

CANCELLA LO SCHERMO = ▯

Dopo aver cambiato le righe del programma, fallo girare per vedere la differenza.

```
 10 REM FIAMMIFERI
 30 E=0
 40 Z=16+INT(8*RND(1))
 50 IF 2*INT(Z/2)=Z THEN 40
 60 H=3+INT(2*RND(1))
 70 PRINT "♥":REM CANCELLA LO SCHERMO
 80 PRINT
 90 PRINT
 110 PRINT "IL NUMERO MASSIMO DI FIAMMIFE
RI CHE PUOIPRENDERE E' ";H
 120 PRINT
 130 IF E>0 THEN PRINT "▶▶▶▶▶▶▶▶HAI PRESO
 ";E;"FIAMMIFERI"
 135 REM CURSORE DESTRO UTILIZZATO 8 VOLT
E
 140 PRINT
 150 IF E>0 THEN PRINT "▶▶▶▶▶▶▶▶HO  PRESO
 ";Q;"FIAMMIFERI"


 160 PRINT
 170 FOR K=1 TO Z
 180 PRINT TAB(15) CHR$(110);" "
 190 IF RND(1)>.6 THEN PRINT
 200 NEXT K
 210 PRINT:PRINT
 220 PRINT "▶▶▶▶▶▶▶▶QUANTI NE VUOI PRENDE
RE ";
 225 REM CURSORE DESTRO UTILIZZATO 8 VOLT
E
 230 INPUT E
 240 IF E>H OR E<1 THEN 230
 250 Z=Z-E
 260 PRINT
 270 IFZ<1 THEN PRINT "HAI PRESO L'ULTIMO
 FIAMMIFERO,HO VINTO!!":END
 290 Q=Z-1-INT((Z-1)/(H+1))*(H+1)
 300 IF Q<1 OR Q>H THEN 290
 310 PRINT
 320 Z=Z-Q
 330 IF Z=0 THEN PRINT "HO PRESO ";Q;"FIA
```

```
MMIFERI "
340 IF Z=0 THEN PRINT "█████████████████HA
I VINTO":END
345 REM CURSORE DESTRO UTILIZZATO 12 VOL
TE
350 GOTO 70
```

Il prossimo cambio che opereremo permettera' di sostituire le informazioni PRINT vuote, che spaziavano le righe, con l'uso del "cursore giu'". Questo simbolo spazia le righe PRINT il numero di volte richiesto, contando verso il basso, partendo dall'ultima riga PRINT stampata o dalla cima dello schermo se e' la prima volta che viene usato.

```
10 REM FIAMMIFERI
30 E=0
40 Z=16+INT(8*RND(1))
50 IF 2*INT(Z/2)=Z THEN 40
60 H=3+INT(2*RND(1))
70 PRINT "█":REM CANCELLA LO SCHERMO
110 PRINT"██IL NUMERO MASSIMO DI FIAMMIF
ERI CHE PUOIPRENDERE E' ";H
115 REM CURSORE GIU'
120 PRINT
130 IF E>0 THEN PRINT "████████████ HAI PR
ESO ";E;"FIAMMIFERI":REM DOWN CURSOR*2
140 REM CURSORE GIU' * 2 VOLTE
150 IF E>0 THEN PRINT "████████████HO  PRE
SO ";Q;"FIAMMIFERI"
155 REM CURSORE GIU'
160 PRINT "██"
165 REM CURSORE GIU' * 2 VOLTE
170 FOR K=1 TO Z
180 PRINT TAB(15) CHR$(110);" ";
190 IF RND(1)>.6 THEN PRINT
200 NEXT K
210 PRINT
220 PRINT "████████████QUANTI NE VUOI PREN
DERE";
225 REM CURSORE GIU' * 2 VOLTE
```

```
230 INPUT E
240 IF E>H OR E<1 THEN 230
250 Z=Z-E
270 IF Z<1 THEN PRINT "HAI PRESO L'UL
TIMO FIAMMIFERO HO VINTO!!":END
290 Q=Z-1-INT((Z-1)/(H+1))*(H+1)
300 IF Q<1 OR Q>H THEN 290
320 Z=Z-Q
330 IF Z=0 THEN PRINT "HO PRESO ";Q;"FIA
MMIFERI"
340 IF Z=0 THEN PRINT "
HAI VINTO":END
345 REM CURSORE GIU' *12 VOLTE
350 GOTO 70
```

Ora che abbiamo fatto girare il programma sullo schermo, usiamo alcuni dei colori del COMMODORE 64. Ecco un'altra versione del programma; questa volta i simboli rappresentano i colori. Le informazioni REM in fine di riga ti dicono quali colori usare. Se non ricordi quali tasti producono i corrispondenti colori, fai riferimento al listato che segue il programma. Quando il colore del testo viene cambiato in questo modo, restera' invariato per il resto del programma fino a che non intervengano altri cambi.

```
10 REM FIAMMIFERI
30 E=0
40 Z=16+INT(8*RND(1))
50 IF 2*INT(Z/2)=Z THEN 40
60 H=3+INT(2*RND(1))
70 PRINT "":REM CANCELLA LO SCHERMO
110 PRINT TAB(4) "IL NUMERO MASSIMO
DI FIAMMIFERI"
113 PRINT TAB(10) "CHE PUOI PRENDERE E'
";H
115 REM PORPORA
130 IF E>0 THEN PRINT "HAI PRE
SO ";E;"FIAMMIFERI":REM ARANCIONE
150 IF E>0 THEN PRINT "HO PRESO
";Q;"FIAMMIFERI":REM VERDE
```

```
160 PRINT "■▒▒":REM NERO
170 FOR K=1 TO Z
180 PRINT TAB(15) CHR$(110);" ";
190 IF RND(1)>.6 THEN PRINT
200 NEXT K
210 PRINT
220 PRINT "▒▒▒▶▶▶▶▶▶▶▶▶▶QUANTI NE VUOI PRE
NDERE ";:REM GIALLO
230 INPUT E
240 IF E>H OR E<1 THEN 230
250 Z=Z-E
270 IF Z<1 THEN PRINT "▒▒▒▒HAI PRESO L'U
LTIMO FIAMMIFERO,HO VINTO!!":END
271 REM BIANCO
290 Q=Z-1-INT((Z-1)/(H+1))*(H+1)
300 IF Q<1 OR Q>H THEN 290
320 Z=Z-Q
330 IF Z=0 THEN PRINT "HO PRESO ";Q;"FIA
MMIFERI"
340 IF Z=0 THEN PRINT "▒▒▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶
HAI VINTO":END
345 REM CURSORE DESTRO UTILIZZATO * 12 V
OLTE
350 GOTO 70
```

```
NERO       =  ■  = CTRL 1

BIANCO     =  ▒  = CTRL 2

ROSSO      =  ▒  = CTRL 3

TURCHESE   =  ◣  = CTRL 4

PORPORA    =  ▒  = CTRL 5

VERDE      =  ▒  = CTRL 6

BLU        =  ▒  = CTRL 7

GIALLO     =  ▒  = CTRL 8
```

```
ARANCIONE = ▚ = COMM 1

MARRONE   = ▘ = COMM 2

ROSSO CH  = ▒ = COMM 3

GRIGIO 1  = ▩ = COMM 4

GRIGIO 2  = ▥ = COMM 5

VERDE CH  = ▌ = COMM 6

AZZURRO   = ▢ = COMM 7

GRIGIO 3  = ▓ = COMM 8
```

Prima di lasciare questo programma, faremo un ultimo cambio. Scrivi questa riga esattamente come viene stampata qui e fai girare di nuovo il programma. Parleremo poi di cio' che e' successo.

```
20 POKE 53280,0:POKE 53281,11
```

POKE E PEEK

Quando abbiamo aggiunto la riga 20 al programma dei fiammiferi, esso ha cambiato il colore dello schermo e del bordo. Il comando POKE richiama una specifica locazione di memoria, identificata dal primo numero e gli assegna un valore. 53280 e' la locazione di memoria che controlla il colore del bordo dello schermo e 53281 e' la locazione che controlla il colore dello schermo. Zero e' il valore del colore nero e 11 e' il valore del colore grigio 1.

Ecco una lista dei valori di tutti i 16 colori:

```
0 @ NERO      8 @ ARANCIONE
1 @ BIANCO    9 @ MARRONE
2 @ ROSSO    10 @ ROSSO CHIARO
3 @ TURCHESE 11 @ GRIGIO 1
4 @ PORPORA  12 @ GRIGIO 2
5 @ VERDE    13 @ VERDE CHIARO
6 @ BLU      14 @ BLU CHIARO
7 @ GIALLO   15 @ GRIGIO 3
```

Per vedere POKE 53280 e POKE 53281 in azione con tutti i 16 colori, stampa questo programma e osserva come il COMMODORE 64 esamina i vari colori:

```
10 REM POKE
20 PRINT "♥":REM CANCELLA LO SCHERMO
30 FOR X=0 TO 15
40 FOR Y=15 TO 0 STEP -1
50 POKE 53280,X
60 POKE 53281,Y
70 FOR T=1 TO 250
80 NEXT T
90 NEXT Y
100 NEXT X
```

Se il comando POKE ancora non ti e' ben chiaro, te lo esemplifichero' maggiormente. Tutte le funzioni nel computer sono controllate dalla memoria incorporata nel computer stesso. Quando il computer viene acceso, lo schermo si presenta blu scuro e il bordo azzurro. Le parti della memoria del computer che controllano cio' sono le locazioni 53281 e 53280. I valori di partenza di queste locazioni sono rispettivamente 6 e 14. Proviamo cio', riportando il computer alla normalita' premendo STOP e RESTORE. Ora che il tuo schermo ha ricuperato i colori originali scrivi:

```
PRINT PEEK(53281) AND 15
```

Il computer stampera' 6. Ora scrivi:

```
PRINT PEEK(53280) AND 15
```

Questa volta la risposta sara' 14. PEEK ti permette di vedere quale valore e' immagazzinato in una locazione di memoria. L'AND 15 mantiene la risposta entro i limiti da noi richiesti. Il metodo per cui il valore di ogni colore e' immagazzinato ci darebbe una risposta confusa se non limitiamo l'intervallo.

Cosi', ora che conosciamo la locazione che immagazzina questa informazione, possiamo cambiarla usando POKE.

Usando POKE abbiamo cambiato il colore dello schermo e del bordo. Quali sono le altre funzioni di POKE?

Fino ad ora, quando abbiamo voluto scrivere informazioni sullo schermo, abbiamo usato PRINT. Cio' inserisce le informazioni sullo schermo in sequenza logica. Questo significa che il computer stampa le righe secondo l'ordine in cui gli vengono presentate.

Tuttavia, usando POKE, possiamo inserire caratteri in qualsiasi zona dello schermo e specificare i loro colori. Lo schermo e' fatto di 25 righe e ogni riga contiene 40 spazi da sinistra verso destra. Percio' lo schermo possiede 1000 locazioni ognuna delle quali e' controllata da una locazione di memoria. L'"indirizzo" dello spazio nell'angolo superiore sinistro dello schermo (il primo), e' 1024 e quello all'angolo inferiore destro (l'ultimo), e' 2023. Cosi', usando questi numeri con POKE, possiamo porre qualsiasi carattere direttamente nella locazione voluta sullo schermo. Ma prima di fare cio', c'e' un'altra cosa da sapere. Queste 1000 locazioni dello schermo possiedono una locazione corrispettiva nella memoria del computer. Questa controlla i colori dei caratteri sullo schermo. Le 1000 locazioni di memoria dei colori partono da 55296 fino a 56295. Cosi' usando POKE con i numeri da 1024 a 2023 e POKE con 55296 a 56295 possiamo porre un carattere direttamente sullo schermo e controllare il suo colore. Scrivi e fai girare questo programma dimostrativo.

```
10 PRINT "□":REM CANCELLA LO SCHERMO
20 POKE 53281,2:POKE 53280,5
30 POKE 1524,83:POKE 1524+54272,7
100 GOTO 100
```

Questo e' un programma senza fine. Premi STOP per riportarlo sotto il tuo controllo.

Dovresti avere uno schermo rosso con un bordo verde e un cuore giallo posto nel mezzo dello schermo. La riga 20 contiene i POKE per lo schermo e il bordo, e la riga 30 per il cuore. Guardiamo piu' attentamente la riga 30. La prima istruzione e' POKE 1524,83. 1524 e' la locazione del centro dello schermo e 83 e' il valore POKE del simbolo cuore. In

ogni modo, puoi stampare qualsiasi carattere con il colore invertito aggiungendo 128 al valore POKE. L' informazione successiva della riga 30 controlla il colore. Il modo piu' semplice per trovare la locazione corrispondente nella memoria dei colori e' di aggiungere la differenza tra i due schermi alla locazione dello schermo. Cioe', 55296 - 1024 e' uguale a 54272. Cosi' 1524 + 54272 ci da' la locazione corretta nella memoria del colore. Il 7 e' il valore del colore giallo. La riga 100 manda il programma alla riga 100. Cio' significa che il programma non finisce e il segno READY non viene stampato sullo schermo. Questo e' un ottimo modo per far si' che sullo schermo compaia solo lo sviluppo del programma.

Aggiungi queste righe al programma per inserire altri simboli sullo schermo.

```
40 POKE 1154,90:POKE 1154+54272,0
50 POKE 1174,88:POKE 1174+54272,5
60 POKE 1874,65:POKE 1874+54272,4
70 POKE 1894,86:POKE 1894+54272,1
```

Ora che abbiamo imparato ad inserire un carattere direttamente sullo schermo in qualsiasi posizione e sappiamo come trovare il valore della locazione di memoria, risulta relativamente facile imparare a muovere i grafici.

Per inserire un carattere sullo schermo e poi muoverlo in maniera logica e' richiesta una formula che controlli la posizione del nostro oggetto mobile. Questa formula e' 1024 + X + Y*40. 1024 e' l'indirizzo di partenza della memoria dello schermo, X e' la posizione orizzontale dell'oggetto sullo schermo e puo' essere un numero da 0 a 39, Y e' la posizione verticale sullo schermo e puo' andare da 0 a 24. Questo e' moltiplicato poi per 40, il numero di spazi orizzontali dello schermo, in modo tale che l'oggetto si possa muovere facilmente su e giu'.

Affinche' l'oggetto mantenga il proprio colore durante il movimento, si usa la seguente formula: 1024 + 54272 + X + Y*40. La formula per avere il valore della locazione dello schermo e' PEEK(1024 + X + Y*40) = n. La n rappresenta il valore che desideri avere.

Ti dimostrero' l'uso di questi con un programma che

inserira' una pallina gialla nel centro di uno schermo rosso e poi muovera' la pallina sullo schermo rimbalzando sui bordi colorati e ad ogni ostacolo presente sul suo cammino.

Scrivi e fai girare il programma e poi parleremo piu' dettagliatamente del listato.

```
10 REM GRAFICI IN MOVIMENTO
20 PRINT "♥":REM CANCELLA LO SCHERMO
30 GOSUB 1000
50 POKE 1024+X+Y*40,81:POKE 1024+54272+X
+Y*40,7
60 FOR DD=1 TO 5:NEXT DD
80 POKE 1024+X+Y*40,32
100 X=X+XX
120 IF PEEK(1024+X+Y*40)=102 THEN XX=-XX
:X=X+XX
140 Y=Y+YY
160 IF PEEK(1024+X+Y*40)=102 THEN YY=-YY
:Y=Y+YY
200 GOTO 50
1000 Y=12:X=20
1005 POKE 53280,0:POKE 53281,2
1010 YY=1:XX=1
1050 FOR S=1024 TO 1063
1060 POKE S,102:POKE S+54272,1
1080 NEXT S
1090 FOR S=1984 TO 2023
1100 POKE S,102:POKE S+54272,1
1110 NEXT S
1120 FOR S=1064 TO 1944 STEP 40
1130 POKE S,102:POKE S+54272,1
1140 NEXT S
1150 FOR S=1103 TO 1983 STEP 40
1160 POKE S,102:POKE S+54272,1
1170 NEXT S
1200 FOR R=1 TO 25
1210 B=INT(1000*RND(1))+1025
1220 IF B=1524 THEN 1210
1230 POKE B,102:POKE B+54272,1
1240 NEXT R
1999 RETURN
```

Le righe 1000 e 1999 mostrano i parametri del programma, X e Y nella riga 1000 sono le coordinate della posizione di partenza della pallina. XX e YY nella riga seguente rappresentano l'ammontare dell'incremento di X e di Y ogni volta che si ripete il ciclo nella parte principale del programma. La riga 1005 cambia i colori dello schermo e dello sfondo e poi le righe da 1010 a 1170 stampano il bordo attorno allo schermo. Il numero 102 e' il valore del simbolo grafico usato per stampare il bordo. Le righe 1200 e 1240 pongono degli ostacoli a caso sullo schermo. Ho usato la riga 1220 per essere sicuro che non sia posto un ostacolo sulla posizione di partenza della pallina.

La linea 50 pone la pallina, con valore 81, sullo schermo e mostra il suo colore. La linea 60 immette una breve pausa in modo tale che la pallina sia vista (cambiando la lunghezza della pausa si altera la velocita' della pallina). In riga 80 noi mettiamo uno spazio vuoto (con valore 32) nello spazio dove si trovava la pallina. Poi si cambia il valore di X incrementandolo con XX. La riga successiva controlla se questa nuova posizione e' occupata dal valore 102, valori del bordo e dell'ostacolo. Se cio' accade, il valore di XX viene cambiato di segno; questo ha l'effetto di cambiare la direzione del movimento della pallina. Aggiungiamo anche il valore di XX ad X. Questo impedisce alla pallina di muoversi nello spazio occupato dall'ostacolo. Questa riga converte XX in un numero negativo in modo tale che venga sottratto da X. Se XX e' gia' negativo, questa riga lo converte in positivo e di conseguenza viene aggiunto ad X. In questo modo possiamo muovere la pallina a sinistra o a destra dello schermo, cambiando il valore di XX da -1 a 1. Questo procedimento e' usato anche per cambiare il valore di X.

Le prossime due righe svolgono le stesse funzioni per la Y. La linea 200 manda il programma alla linea 50 in modo tale che la pallina possa essere inserita nella sua nuova posizione.

Il COMMODORE 64 ha un altro sistema di grafici chiamato SPRITE grafici che comporta la creazione di tuoi personali oggetti grafici movibili sullo schermo.

SPRITE grafici hanno l'abilita' di creare effetti tridimensionali muovendo un oggetto dietro l'altro.

Comunque, siccome lo scopo di questo libro e' di introdurti ai programmi in BASIC, gli SPRITE non rientrano nella nostra discussione.

C A P I T O L O   8

O P E R A T O R I   R E L A Z I O N A L I

Tutti conosciamo il segno uguale (=) e lo abbiamo visto in opera in molti dei precedenti programmi. Abbiamo anche visto il segno "maggiore di" (>), il "minore di" (<) e il "diverso da" (<>). A questo punto del libro, penso sia utile ricapitolare brevemente il significato di questi simboli:

= uguale

> maggiore di

< minore di

>= maggiore o uguale a

<= minore o uguale a

<> diverso da

Nel prossimo programma vedremo in uso questi simboli che ci permetteranno di sfidare il computer in un gioco, GOMOKU, basato su un programma di G. Charlton.

In questo gioco tu e il COMMODORE 64 fate a turno a porre i vostri pezzi in una scacchiera di otto per otto. Lo scopo del gioco e' di inserire quattro pezzi in una riga, verticalmente, orizzontalmente o diagonalmente. I pezzi del computer sono dei cerchietti neri e i tuoi dei cerchietti rossi. Scoprirai che il computer ha una difesa molto buona.

All'inizio del gioco ti verra' offerta l'opportunita' di muovere per primo. Per porre un pezzo sulla scacchiera, inserisci il numero del quadratino in cui tu desideri mettere il tuo pezzo. Inseriscilo come un numero di due cifre in cui la prima cifra e' la coordinata verticale della scacchiera seguita da quella orizzontale. Per esempio, le coordinate di un quadratino posto sulla prima riga della scacchiera possono essere 14.

Ecco alcune mosse del gioco. Per rendere piu' visibile la differenza tra i pezzi del computer e i tuoi, abbiamo

inserito "h" per i tuoi pezzi e "c" per quelli del computer. Dopo gli esempi vi mostreremo il listato.

```
  12345678
1 ........ 1
2 ..H..... 2
3 ..H..... 3
4 ........ 4
5 ........ 5
6 ........ 6
7 ........ 7
8 C....... 8

  12345678
```

```
  12345678
1 ........ 1
2 ..H..... 2
3 ..H..... 3
4 ..C..... 4
5 ........ 5
6 ........ 6
7 ........ 7
8 C....... 8

  12345678
```

```
  12345678
1 ........ 1
2 ..H..... 2
3 ..HH.... 3
4 ..C..... 4
5 ........ 5
6 ........ 6
7 ........ 7
8 C....... 8

  12345678
```

```
10 REM GOMOKU
15 POKE,53281,11:POKE 53280,0
20 GOSUB 1180
30 GOSUB 230
60 GOSUB 420
70 GOSUB 230
80 GOSUB 540
90 GOSUB 230
100 GOSUB 1030
110 GOTO 60
140 E=A
145 E=E+N:IF A(E)<>Z THEN RETURN
150 K=K+1:GOTO 145
230 PRINT:PRINT "▒":PRINT TAB(14) "▮▮▮▮▮
▮ 12345678":REM HOME
235 REM CURSORE GIU' UTILIZZATO * 5 VOLT
E:REM AZZURRO
240 FOR A=1 TO 8:PRINT "▮" TAB(12) A;:RE
M AZZURRO
270 FOR B=2 TO 9:M=A(A*10+B)
280 IF M=C THEN PRINT "■●";:REM NERO
290 IF M=H THEN PRINT "▒●";:REM ROSSO CH
IARO SHIFT Q
300 IF M=46 THEN PRINT "▮.";:REM AZZURRO
310 NEXT:PRINT "▮" A:NEXT:REM AZZURRO
315 PRINT
320 PRINT TAB(14) " 12345678▮▮":REM CURS
ORE GIU' UTILIZZATO * 2 VOLTE
360 RETURN
420 PRINTTAB(13) "LA TUA MOSSA";:INPUT G
:G=G+1:IFG<12 OR G>89 OR A(G)<>46 THEN42
0
460 PRINT "▒":REM HOME CURSOR
470 PRINT TAB(23) "▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮     "
475 REM CURSORE GIU' UTILIZZATO * 19 VOL
TE 5 SPAZI
480 Z=H:A(G)=Z:RETURN
540 A=G:L=0:FOR X=1 TO 4:K=0:N=X(X)
590 GOSUB 140
600 N=-N:GOSUB 140
610 IF K>L THEN L=K
```

```
630 NEXT
640 IF L>3 THEN PRINT TAB(14) "████HAI V
INTO!!":END
645 REM CURSORE GIU' UTILIZZATO * 4 VOLT
E
650 T=1
660 IF T<>2 THEN Z=C
670 IF T=2 THEN Z=H
680 G=0:H1=0:L=0
710 FOR A=12 TO 89:M=0:IF A(A)<>46 THEN
900
740 FOR X=1 TO 4:K=0:N=X(X):GOSUB 140
780 N=-N:GOSUB 140
800 IF K>L THEN H1=0:L=K
810 IF L<>K THEN 860
820 IF T=1 AND L<4 OR(T=2 OR T=3) AND L<
2 THEN 860
850 M=M+1
860 NEXT
870 IF M<H1 THEN 900
880 H1=M:G=A
900 NEXT
910 IF H1<>0 THEN 980
920 T=T+1:IF T<>4 THEN 660
940 A=1
950 G=INT(RND(1)*77)+13
960 IF A(G)=46 THEN 980
970 A=A+1:IF A<400 THEN 950
975 PRINT TAB(13) "████ MI ARRENDO !!":R
EM CURSORE GIU' UTILIZZATO * 4 VOLTE
980 A(G)=C:RETURN
1030 Z=C:A=G:L=0
1060 FOR X=1 TO 4:K=0:N=X(X)
1090 GOSUB 140
1100 N=-N:GOSUB 140
1120 IF K>L THEN L=K
1130 NEXT
1140 IF L>3 THEN PRINT TAB(14) "████ HO
VINTO !!":END
1145 REM CURSORE GIU' UTILIZZATO * 4 VOL
TE
```

```
1150 RETURN
1180 PRINT "♥"
1200 DIM A(100)
1210 FOR A=1 TO 8:FOR B=2 TO 9:A(A*10+B)
=46:NEXT B,A
1260 FOR Q=1 TO 4:READ X(Q):NEXT
1290 DATA 1,9,10,11
1300 H=ASC("H"):C=ASC("C")
1340 PRINT:PRINT TAB(2) "INSERISCI (S) S
E VUOI LA PRIMA MOSSA"
1345 PRINT TAB(6) "INSERISCI (N) SE NON
LA VUOI"
1350 GET F$:IF F$<>"N" AND F$<>"S" THEN
GOTO 1350
1380 PRINT "♥"
1390 IF F$="S" THEN RETURN
1400 FOR J=1 TO INT(RND(1)*11)+1
1410 READ Z:NEXT
1440 A(Z)=C:RETURN
1450 DATA 34,35,36,44,46,47,54,55,56,57,
66
```

Nel programma GOMOKU i simboli di comparazione hanno una grande importanza. Essi generano un numero di tests insieme alle istruzioni IF/THEN e osservano quali caratteri stampare (righe 280-300), controllano se il computer ha vinto (linea 1140), trovano risposta alle domande poste all'inizio del gioco (righe 1340-1390) e svolgono molti altri compiti.

Le parole AND e OR sono usate anche nelle linee di comparazione, legando i tests insieme, come puoi vedere in riga 820. Queste due parole lavorano in questo modo:

AND il computer fa cio' che segue il THEN se entrambe le condizioni legate dall' AND sono vere.
OR il computer esegue l'istruzione che segue il THEN se almeno una delle condizioni e' vera.

Guardiamo la logica della riga 820. Il computer guarda se T e' uguale a 1 e L e' minore di 4 o se T e' uguale a 2 o a

3 e L e' minore di 2. Se una di queste due condizioni e' vera, allora il programma va in linea 860.

C A P I T O L O 9

D U E G I O C H I E U N T E S T

E' ora di fare una pausa e di divertirci un po' con il nostro COMMODORE 64. Come potrai vedere, in questo capitolo ci sono tre programmi che usano dei comandi non ancora spiegati. Ti consiglio di inserire i programmi cosi' come sono, giocarci e poi tornare sulle spiegazioni che seguono i listati, dopo che avrai esaminato l'intero testo.

Comunque se preferisci continuare nello studio dei comandi del computer, passa oltre.

SOLITARIO

Il nostro primo listato ti permette di usare il computer come una scacchiera.

Lo scopo di questo gioco e' semplice da spiegare, ma non e' cosi' semplice da attuare. Si comincia con una scacchiera che ha 33 "buchi" disposti a forma di croce. In 32 di questi buchi ci sono delle biglie, mentre la posizione centrale e' vuota.

Quando fai girare il programma, comparira' la scacchiera sullo schermo e ti sara' mostrato il numero delle mosse fatto fino a quel momento. Per giocare, devi semplicemente saltare uno qualsiasi dei tuoi pezzi, verticalmente o orizzontalmente, in modo tale da finire in uno spazio vuoto. Il pezzo che hai saltato viene sottratto dalla scacchiera. Lo scopo di questo gioco e' finire con un solo pezzo nella posizione centrale.

Ti si dice il numero della mossa, e quanti pezzi sono rimasti sulla scacchiera. Tu muovi facendo entrare le coordinate del pezzo che vuoi spostare, usando per prima la coordinata verticale, seguita da quella orizzontale. Queste vengono inserite come un solo numero di due cifre.

Per esempio, se vuoi muovere il pezzo che si trova in due posizioni sotto il buco centrale all'inizio del gioco,

dovrai inserire 64, poi premere RETURN, seguito da 44, e ancora RETURN. La scacchiera verra' cosi' nuovamente stampata e ti verra' offerta la possibilita' di fare un'altra mossa.

Ecco il listato del programma SOLITARIO.

```
10 REM SOLITARIO
15 POKE 53280,14:POKE 53281,7
20 GOSUB 400
30 GOSUB 250
40 REM MOSSA ACCORDATA
50 PRINT TAB(5) "SU QUALE PIOLO DESIDERI
 MUOVERE";
60 INPUT A
70 IF A=99 THEN GOTO 240
80 IF A<11 OR A>77 THEN GOTO 50
90 IF A(A)<>79 THEN GOTO 50
100 PRINT TAB(5) "SU ";A;"DA DOVE";
110 INPUT B
120 IF B<11 OR B>77 THEN GOTO 110
130 IF A(B)<>E THEN GOTO 110
140 A((A+B)/2)=E:A(A)=E:A(B)=79
150 YT=YT+1
160 C=0
170 FOR F=11 TO 75
180 IF A(F)=79 THEN C=C+1
190 NEXT F
200 GOSUB 250
210 PRINT TAB(5) "CI SONO";C;"PIOLI SULL
'ASSE":PRINT
220 IF C<>1 THEN GOTO 40
230 IF A(44)=79 THEN PRINT "♥":PRINT "L'
HAI FATTO IN";YT;"MOSSE":END
235 REM CANCELLA LO SCHERMO
240 PRINT "♥":PRINT "IL GIOCO E' FINITO
E HAI SBAGLIATO":END
245 REM CANCELLA LO SCHERMO
250 REM STAMPA ASSE
260 PRINT "♥":REM CANCELLA LO SCHERMO
270 PRINT TAB(5) "■INSERISCI PER PRIMA L
E COORDINATE"
```

```
273 PRINT "        DELLA POSIZIONE DA TE SC
ELTA":PRINT
275 REM GRIGIO 1
280 PRINT TAB(5) "INSERISCI 99 SE VUOI A
RRENDERTI"
290 PRINT TAB(10) "1 2 3 4 5 6 7"

300 PRINT TAB(10);
310 FOR D=11 TO 75
320 T=10*(INT(D/10))
330 IF D-T=8 THEN D=D+2:PRINT T/10:PRINT
 TAB(10);:GOTO 350
340 PRINT CHR$(A(D));" ";
350 NEXT D:PRINT "    7"
360 PRINT:PRINT:PRINT
370 PRINT TAB(5) "MOSSE FATTE FINO AD OR
A:-";YT:PRINT
380 PRINT:PRINT
390 RETURN
400 REM INIZIALIZZAZIONE
410 PRINT "⅃":REM CANCELLA LO SCHERMO
420 DIM A(87)
430 E=42
440 FOR D=11 TO 75
450 T=10*(INT(D/10))
460 IF D-T=8 THEN D=D+3
470 READ A(D)
480 NEXT D
490 YT=0

500 RETURN
510 REM 42 IS ASC("*")
520 REM 79 IS ASC("O")
530 DATA 32,32,79,79,79,32,32
540 DATA 32,32,79,79,79,32,32
550 DATA 79,79,79,79,79,79,79
560 DATA 79,79,79,42,79,79,79
570 DATA 79,79,79,79,79,79,79
580 DATA 32,32,79,79,79,32,32
590 DATA 32,32,79,79,79
```

La linea 20 manda l'azione alla subroutine partente dalla riga 400 che inizializza le variabili, dopo aver cancellato lo schermo in riga 410. L'array A e' dimensionato in riga 420.

E e' il valore di un quadratino vuoto. Puoi vedere qual'e', scrivendo PRINT CHR$(2). Il ciclo da 470 a 500 carica molti degli elementi dell'array prendendoli dalle istruzioni DATA contenute nelle righe da 530 a 590.

Ritornando dalla routine di inizializzazione, il programma dirige il computer alla routine che parte dalla riga 250, che stampa la scacchiera. Di seguito viene eseguita la parte principale del programma (da 40 a 150), accettando la tua mossa, e operando i cambi necessari agli elementi dell'array A.

La routine che va dalla riga 160 alla riga 220 esamina la scacchiera usando la variabile C per contare i pezzi rimasti sulla scacchiera. La riga 200 chiama la subroutine addetta a stampare la scacchiera sullo schermo. La riga 210 ti segnala quanti pezzi sono rimasti e (in riga 220) se e' rimasto piu' di un pezzo sulla scacchiera, il gioco torna alla riga 40 per un altro turno.

Se C e'uguale a 1 in riga 220, cio' significa che e' rimasta una sola biglia sulla scacchiera; il programma allora salta il GOTO in riga 220 e va a finire il gioco. Il computer infatti verifica (riga 230) se l'elemento dell'array che contiene la biglia e' 44; se cio' e' vero, allora il computer capisce che hai vinto e stampa un messaggio di congratulazioni. Altrimenti la linea 240 fa apparire questo messaggio: "Il gioco e' finito e tu hai fallito".

PROVA DI RIFLESSI

Il prossimo programma e' molto piu' corto del SOLITARIO, ma piu' divertente. Inserisci il programma, scrivi RUN, e il messaggio ATTENDI apparira'. Dopo una breve attesa, le parole "OK PREMI IL TASTO Z" comparira' sullo schermo. Cerca il piu' velocemente possibile il tasto Z e premilo, intanto che il computer contera' il tempo di reazione.

Il computer ti dira' poi in quanto tempo hai reagito e paragonera' il tuo risultato al miglior risultato

precedente. "IL MIGLIORE FINO AD ORA E'..." apparira' sullo schermo, e il computer aspettera' che tu tolga le mani dalla tastiera prima di ricominciare ancora.

ATTENDI

OK PREMI IL TASTO Z

IL TUO PUNTEGGIO ERA 25

IL MIGLIORE FINO AD ORA E' 27

Il gioco continuera' fino a che tu non riuscirai ad avere un tempo di reazione al di sotto di 20, compito davvero non facile. Ecco il listato del programma:

```
10 REM PROVA DEI RIFLESSI
20 HS=1000
30 POKE 53280,2:POKE 53281,10
50 PRINT "□":REM CANCELLA LO SCHERMO
60 PRINT:PRINT:PRINT TAB(14) "■ATTENDI":
REM GIALLO
70 FOR A=1 TO 1000+INT(3000*RND(1))
80 NEXT A
90 GET A$
100 IF A$<>"" THEN GOTO 70
110 PRINT:PRINT:PRINT:PRINT TAB(9) "■OK
PREMI IL TASTO Z":REM BIANCO
120 C=0
130 C=C+1
140 GET A$
150 IF A$<>"Z" THEN GOTO 130
160 PRINT:PRINT:PRINT:PRINT TAB(7) "■IL
TUO PUNTEGGIO ERA";C:REM TURCHESE
170 IF C<HS THEN HS=C
180 PRINT:PRINT TAB(5) "■IL MIGLIORE FIN
```

```
0 AD ORA E'";HS:REM BLU
190 FOR A=1 TO 1500:NEXT A
200 GET A$
210 IF A$<>"" THEN GOTO 200
220 IF HS>20 THEN GOTO 50
230 PRINT "♫":REM CANCELLA LO SCHERMO
240 PRINT:PRINT:PRINT:PRINT:PRINT
250 PRINT TAB(10) "SEI IL CAMPIONE!!":EN
D
```

La linea 20 pone la variabile HS uguale a 1000. La variabile C e' posta uguale a zero in riga 120 ed incrementata di uno ogni volta che si passa attraverso la riga 130; cio' accade quando non riesci a premere il tasto Z. Le righe 140 e 150 controllano che tu abbia premuto il tasto Z, altrimenti mandano il programma alla riga 70 dove C viene incrementato.

Una volta premuto il tasto Z il programma va in riga 160 dove ti viene detto il tuo punteggio. Questo viene poi comparato con il miglior punteggio precedente (variabile HS) nella riga seguente; se C e' minore di HS allora HS diventera' uguale a C.

La linea successiva (190) immette una breve pausa, e poi la riga 200 controlla che tu abbia levato le mani dalla tastiera. Il programma gira fra le righe 200 e 210 fino a che tu non tolga le mani dai tasti. Il GOTO poi manda il programma indietro fino alla riga 50 e comincia il prossimo turno.

Il GOTO continua solo se HS rimane maggiore di 20 (come puoi veder in riga 220). Se hai ottenuto un punteggio al di sotto di 20 il programma continua attraverso la riga 220 per le righe 230, 240 e 250 la quale fara' apparire sullo schermo le parole "SEI IL CAMPIONE!". La riga 30 cambia il colore del bordo e dello schermo.

HASAMI SHOGI

Il nostro prossimo programma e' molto interessante. Ci permette di giocare contro il computer nel gioco giapponese HASAMI SHOGI. Malgrado tutte le scacchiere a noi familiari

siano di otto per otto, HASAMI SHOGI si gioca su una scacchiera di nove per nove.

Ogni giocatore inizia il gioco con 18 "pietre". Tu inizi dalla parte inferiore della scacchiera, mentre il computer avra' i suoi pezzi posti in cima allo schermo. Lo scopo del gioco e' quello di catturare sette pezzi dell'avversario. Veramente, lo scopo del gioco sarebbe di catturare tutti i pezzi dell'avversario, ma, come potrai verificare quando giocherai, questo renderebbe il gioco troppo lungo.

Puo' essere soddisfacente quando si gioca contro un altro essere umano, ma ti accorgerai che diventa veramente lungo completare il gioco contro il computer. Percio' ne abbiamo cambiato lo scopo per renderlo piu' corto e di conseguenza piu' divertente.

Puoi muovere un solo pezzo alla volta, verticalmente o orizzontalmente, ma non diagonalmente. Per ogni mossa hai tre possibilita':

1. Puoi muovere in un quadratino vuoto trovantesi sopra, sotto o accanto al tuo pezzo.

2. Puoi saltare uno dei tuoi pezzi e arrivare in un quadratino vuoto.

3. Puoi saltare un pezzo del tuo avversario e arrivare in un quadratino vuoto.

Al contrario della dama, un pezzo saltato non e' catturato, quindi non viene sottratto dalla scacchiera. L'unico modo per catturare un pezzo consiste nel muoverti in modo tale che il tuo pezzo accalappi un pezzo del computer tra due dei tuoi; esso cattura i tuoi pezzi nello stesso modo.

Ma tu non perdi il tuo pezzo semplicemente se viene a trovarsi tra due pezzi del computer, percio' il tuo scopo sara' di avere sempre un quadratino vuoto vicino per poterlo sfruttare nella mossa successiva. Osservare il gioco e' forse il miglior modo per capirlo. Ecco alcuni fasi del gioco HASAMI SHOGI giocato contro il COMMODORE 64:

PREMI UN TASTO QUALSIASI

```
  1 2 3 4 5 6 7 8 9
I C C C C C C C C C I
H C C C C C C C C C H
G * * * * * * * * * G
F * * * * * * * * * F
E * * * * * * * * * E
D * * * * * * * * * D
C * * * * * * * * * C
B H H H H H H H H H B
A H H H H H H H H H A
  1 2 3 4 5 6 7 8 9

COMPUTER:  0

UOMO:  0
```

```
  1 2 3 4 5 6 7 8 9
I C C C C C C C C C I
H C C C C C C * C C H
G * * * * * * C * * G
F * * * * * * * * * F
E * * * * * * * * * E
D * * * * * * * * * D
C * * * * * * H * * C
B H H H H H H * H H B
A H H H H H H H H H A
  1 2 3 4 5 6 7 8 9

COMPUTER:  0

UOMO:  0
```

```
  1 2 3 4 5 6 7 8 9
I C C C C C C C C C I
H C C C C C C * C C H
G * * * * * * C * * G
F * * * * * * * * * F
E * * * * * * * * * E
D * * * * * * * * * D
C * * * * * * * * * C
B H H H H H H H H H B
A H H H H H H H H H A
  1 2 3 4 5 6 7 8 9

COMPUTER:  0

UOMO:  0

DA (LETTERA,NUMERO)
```

```
  1 2 3 4 5 6 7 8 9
I C C C C C C C C C I
H C C * C C C * C C H
G * * C * * * C * * G
F * * * * * * * * * F
E * * * * * * * * * E
D * * * * * * * * * D
C * * * * * * H * * C
B H H H H H H * H H B
A H H H H H H H H H A
  1 2 3 4 5 6 7 8 9

COMPUTER:  0

UOMO:  0

DA (LETTERA,NUMERO)
```

```
10 REM HASAMI SHOGI
20 POKE 53280,11
30 GOSUB 790
40 GOSUB 90
50 GOSUB 460
60 GOSUB 630
70 GOSUB 460
80 GOTO 40
90 REM CATTURA
100 A=99
110 IF A(A)<>C THEN 190
120 IF A(A-10)=E THEN IF A(A-9)=H THEN I
F A(A-8)=C THEN B=A-10:GOTO 350
130 IF A(A-10)=E THEN IF A(A-11)=H THEN
IF A(A-12)=C THEN B=A-10:GOTO 350
140 IF A(A-10)=E THEN IF A(A+11)=H THEN
IF A(A+12)=C THEN B=A-10:GOTO 350
150 B=1
160 IF A+2*C(B)<11 OR A+2*C(B)>99 THEN G
OTO 180
170 IF A(A+C(B))=E AND A(A+2*C(B))=H AND
 A(A+3*C(B))=C THEN A(A+2*C(B))=E
175 IF A(A+C(B))=E AND A(A+2*C(B))=H AND
 A(A+3*C(B))=C THEN CS=CS+1:GOTO 340
180 IF B<4 THEN B=B+1:GOTO 160
190 IF A>11 THEN A=A-1:GOTO 110
200 REM NON CATTURA
210 CO=0
220 CO=CO+1
230 A=INT(89*RND(1))+1
240 IF A(A)=C THEN 270
250 IF CO<200 THEN 220
260 PRINT TAB(6) "CAPO AMMETTO LA SCONFI
TTA!!":END
270 B=1
280 IF A+2*C(B)<11 THEN 300
290 IF (A(A+C(B))=C OR A(A+C(B))=H) AND
A(A+2*C(B))=E THEN B=A+2*C(B):GOTO 350
300 IF A(A+C(B))=E THEN 330
310 IF B<4 THEN B=B+1:GOTO 280
320 GOTO 250
```

```
330 REM MOSSE DEL COMPUTER
340 B=A+C(B)
350 B1=B-10*(INT(B/10))
360 A(B)=C:A(A)=E
370 IF B1>7 THEN GOTO 390
380 IF A(B+1)=H AND A(B+2)=C THEN A(B+1)
=E:CS=CS+1
390 IF B1<3 THEN 410
400 IF A(B-1)=H AND A(B-2)=C THEN A(B-1)
=E:CS=CS+1
410 IF A>89 THEN GOTO 430
420 IF A(B+10)=H AND A(B+20)=C THEN A(B+
10)=E:CS=CS+1
430 IF A<29 THEN RETURN
440 IF A(B-10)=H AND A(B-20)=C THEN A(B-
10)=E:CS=CS+1
450 RETURN
460 REM GRAFICO SULLO SCHERMO
470 PRINT "□":PRINT:PRINT:REM CANCELLA L
O SCHERMO
480 PRINT TAB(12) "▥1 2 3 4 5 6 7 8 9"
490 FOR M=90 TO 10 STEP -10
500 PRINT TAB(10) CHR$(M/10+64);" ";
510 FOR N=1 TO 9
520 PRINT CHR$(A(M+N));" ";
530 NEXT
540 PRINT CHR$(M/10+64)
550 NEXT
560 PRINT TAB(12) "1 2 3 4 5 6 7 8 9"
570 PRINT:PRINT TAB(10) "▩COMPUTER: ";CS
:REM ROSSO CHIARO
580 PRINT:PRINT TAB(10) "UOMO: ";HS
590 IF CS>6 OR HS>6 THEN GOTO 610
600 RETURN
610 IF CS>HS THEN PRINT:PRINT TAB(15) "H
O VINTO!!!":END
620 PRINT:PRINT TAB(14) "HAI VINTO!!!":E
ND
630 REM MOSSA DELL'UOMO
640 PRINT:PRINT TAB(8) "◣DA (LETTERA,NUM
ERO)";:REM TURCHESE
```

```
645 INPUT A$
650 IF A$="S" THEN END
660 IF LEN(A$)<>2 THEN 640
670 PRINT TAB(8) "▒DA ";A$;" A";:INPUT B
$:REM PORPORA
680 IF LEN(B$)<>2 THEN 670
690 A=10*(ASC(A$)-64)+VAL(RIGHT$(A$,1))
700 B=10*(ASC(B$)-64)+VAL(RIGHT$(B$,1))
710 Y=VAL(RIGHT$(B$,1))
720 A(B)=H:A(A)=E
730 IF A(B+1)=C AND A(B+2)=H AND Y<=7 TH
EN A(B+1)=E:HS=HS+1
740 IF A(B-1)=C AND A(B-2)=H AND Y>=3 TH
EN A(B-1)=E:HS=HS+1
750 IF B>79 THEN GOTO 670
760 IF A(B+10)=C AND A(B+20)=H THEN A(B+
10)=E:HS=HS+1
780 RETURN
790 REM INIZIALIZZAZIONE
800 PRINT "▯":REM CANCELLA LO SCHERMO
810 PRINT:PRINT:PRINT TAB(8) "PREMI UN T
ASTO QUALSIASI"
840 GET A$:IF A$="" THEN 840
870 PRINT "▯":REM CANCELLA LO SCHERMO
880 DIM A(129):DIM C(4)
890 H=72:C=67:E=42
900 FOR Z=11 TO 29
910 IF Z=20 THEN Z=21
920 A(Z)=H
930 NEXT
940 FOR Z=31 TO 79
950 IF 10*INT(Z/10)=Z THEN Z=Z+1
960 A(Z)=E
970 NEXT
980 FOR Z=81 TO 99
990 IF Z=90 THEN Z=91
1000 A(Z)=C
1010 NEXT
1020 HS=0
1030 CS=0
1040 FOR Z=1 TO 4
```

```
1050 READ C(Z)
1060 NEXT
1070 DATA -10,-1,1,10
1080 GOSUB 460
1090 RETURN
```

Questo listato sembra molto lungo, ma e' semplice da capire. Questo accade perche' e' scritto in 'moduli', ognuno dei quali e' chiamato da un GOSUB all'inizio del programma. Programmare in questo modo e' molto utile se stai usando dei programmi lunghi, perche' ti permette di saper esattamente cosa svolge ogni sezione.

Nello stesso modo, e' facile evidenziare i 'bugs' (la parola nel gergo del computer significa errore). Il modulo che contiene l'errore verra' isolato con molta facilita'.

Ecco come si svolge cio' in questo programma.

Riga 30: GOSUB 790. Questo manda il programma alla subroutine che fissa tutte le variabili di partenza. Questo tipo di subroutine e' spesso chiamata subroutine di inizializzazione.

Riga 40: GOSUB 90. Questa e' la subroutine che determina la mossa del computer.

Riga 50: GOSUB 460. La subroutine che parte dalla riga 460 e ristampa la scacchiera dopo ogni mossa.

Riga 60: GOSUB 630. Questa subroutine accetta ed esegue la mossa del giocatore.

Riga 70: viene chiamata ancora la subroutine per stampare la scacchiera.

Riga 80: GOTO 40. Questo rimanda l'azione all'inizio del ciclo. Ora che sappiamo la funzione di ogni modulo, possiamo facilmente trovare un eventuale errore.

Siccome le condizioni per terminare il gioco sono all'interno di altri moduli, esso continuera' a girare in questo modo fino a che non si incontrera' con una di queste condizioni. Puoi capire da cio' che se, ad esempio, il computer compie degli errori di stampa sulla scacchiera, sara' facile trovare la subroutine cominciante in riga 460, perche' e' proprio questa che svolge la funzione suddetta.

Nello stesso modo, se il computer ha mosso, per esempio, il pezzo dove avevi segnalato di volerti muovere, sara'

facile individuare l'errore nella subroutine che si occupa della tua mossa, cioe' quella che parte dalla riga 630.

La possibilita' di riconoscere dove e' situato l'errore nel tuo programma in breve tempo, risulta di grande utilita'.

E' possibile, in un programma come questo, che i veri problemi si trovino nelle routines che danno "intelligenza" al computer. Ho scoperto che il miglior modo per lavorare e' scrivere le prime righe (per esempio le righe dalla 10 alla 80 in questo programma) prima che venga scritta qualsiasi altra cosa. Poi, scrivo le altre parti modulo per modulo. Una volta terminata l'inizializzazione, posso lavorare sulle routines che stampano la scacchiera, senza preoccuparmi di tutto il resto.

Cosi' posso scrivere gli altri moduli, uno per uno, senza preoccuparmi dei precedenti.

Scoprirai che puoi seguire questo metodo per qualsiasi programma tu debba scrivere. Anchi i programmi commerciali possono essere trattati nello stesso modo. Dividi il programma in moduli a se' stanti e poi scrivi all'inizio del progrmma tutte le relative chiamate. Sebbene questo renda i programmi piu' lunghi, il guadagno nel comprendere piu' facilmente supera la scrittura aggiuntiva necessaria.

C A P I T O L O 10

L E S T R I N G H E

Abbiamo nominato le variabili numeriche (lettere come A o B, o combinazioni come R2 o C3) e le variabili stringa (una o piu' lettere seguite dal segno del dollaro, come NA$, A$ o AG$) svariate volte nei precedenti capitoli. In questo esamineremo le stringhe e cio' che si puo' fare con il loro aiuto.

I CARATTERI

Ogni lettera, numero o simbolo che il COMMODORE 64 stampa, ha un codice (il codice ASCII, parola che verra' spiegata nel glossario). Dicendo al computer di stampare il carattere del codice, esso riproduce il carattere stesso.

E' facile da capire. Siccome il codice e' ASCII, il comando del computer si chiama ASC. Nota che il valore ASC per la lettera "A" non ha niente a che fare con il valore assegnato ad "A" come variabile numerica, ma fa riferimento ad "A" quando si vuole che il computer stampi la lettera "A". Ti sarai anche accorto che abbiamo messo la lettera "A" tra virgolette quando ci riferiamo ad essa trattandola come lettera.

Prova adesso. Inserisci la seguente istruzione nel computer ed osserva che cosa accade:

```
PRINT ASC("A")
```

Nota che la lettera non solo deve essere tra virgolette, ma anche tra parentesi. Quando avrai fatto girare il programma, avrai ottenuto la risposta 65. (Se non e' cosi', hai dimenticato qualcosa nella riga).

Da cio' si puo' vedere che 65 e' l'ASC di "A". Possiamo convertire il 65 nella lettera "A" chiedendo al computer di stampare il carattere che corrisponde al codice ASC 65. Cio' e' fattibile con la parola in BASIC CHR$, come segue:

```
PRINT CHR$(65)
```

Fai girare, e la lettera "A" apparira'. Puoi fare in modo che il tuo computer stampi tutti i codici ASC e i suoi caratteri corrispondenti mediante questo breve programma. Inseriscilo e osserva attentamente:

```
10 REM DIMOSTRAZIONE DI ASC E CHR$
20 FOR A=32 TO 255
30 PRINT A;CHR$(A);" ";
40 FOR B=1 TO 100:NEXT B
50 NEXT A
```

Questa e' la prima parte del programma che apparira':

```
 32    33 !  34 "  35#  36$  37%  38
&  39'  40(  41)  42*  43+  44,  4
5-  46.  47/  480  491  502  513
 524  535  546  557  568  579  58
:  59;  60<  61=  62>  63?  64@  6
5A  66B  67C  68D  69E  70F  71G
 72H  73I  74J  75K  76L  77M  78
N  79O  80P  81Q  82R  83S  84T  8
5U  86V  87W  88X  89Y  90Z  91[
 92£  93]  94↑  95←  96─  97♠  98
│  99─  100─  101─  102─  103│  104
│  105╮  106╮  107╯  108└  109╲  1
10╱  111┌  112┐  113●  114─  115♥
 116│  117╭  118╳  119○  120♣  121
│  122♦  123┼  124▒  125│  126π  12
7◥  128▬  129▬  130▬  131▬  132▬
```

Nota che i codici ASCII sono differenti dai valori POKE.

PROVA IL TUO CARATTERE

Il prossimo programma e' un test di reazione come quello precedente. Questa volta pero' devi cercare di trovare il tasto giusto sulla tastiera il piu' velocemente possibile.

Una lettera apparira' sullo schermo; tu cercala sulla tastiera e premi il tasto corrispondente. Ti verra' detto quanto tempo avrai impiegato, e questo tempo sara' paragonato al tuo miglior tempo.

Nota che la lettera stampata sullo schermo usa CHR$ in riga 70 stampando il carattere di un numero a caso (in riga 40) e assegnato ad una variabile A. A$ compare in linea 80 insieme a GET (che verra' spiegato piu' tardi) e verra' paragonato alla lettera che il computer ha scelto, in riga 90.

```
10 REM PROVA IL TUO CARATTERE
20 PRINT "♥":REM CANCELLA LO SCHERMO
30 BE=1000
40 A=65+INT(26*RND(1))
50 B=0
60 PRINT:PRINT:PRINT:PRINT
70 PRINT TAB(18) CHR$(A)
80 GET A$:PRINT "■":PRINT:PRINT:PRINT:PR
INT:REM CURSOR HOME
90 IF A$=CHR$(A) THEN GOTO 160
100 B=B+1
110 PRINT:PRINT:PRINT:PRINT:PRINT
120 PRINT TAB(17) B
130 IF B<200 THEN GOTO 80
140 PRINT:PRINT:PRINT TAB(7) "SPIACENTE
IL TEMPO E' FINITO"
150 GOTO 180

160 PRINT:PRINT:PRINT:PRINT TAB(4) "BEN
FATTO!IL TUO PUNTEGGIO E' ";B
180 IF B<BE THEN BE=B
190 PRINT:PRINT:PRINT:PRINT
200 PRINT TAB(1) "IL MIGLIOR PUNTEGGIO F
INO AD ORA E'";BE
210 FOR G=1 TO 50*B
220 NEXT G
230 PRINT "♥":REM CANCELLA LO SCHERMO
240 GOTO 40
```

MANIPOLAZIONI DI STRINGHE

Un aspetto veramente utile del BASIC sul tuo COMMODORE 64 e' il modo usato per manipolare le stringhe. Le parole usate sono:

LEFT$
MID$
RIGHT$

(Nel gergo parlato queste vengono chiamate: stringa di sinistra, stringa di centro e stringa di destra).

Il prossimo programma le mostra in azione. Inseriscilo e fallo girare sul tuo computer, poi riprendi il libro per la spiegazione.

```
10 PRINT"♥":REM CANCELLA LO SCHERMO
20 A$="QUINTA*STRADA"
30 PRINT
40 PRINT "LEFT$(A$,3) = ";LEFT$(A$,3)
50 PRINT
60 PRINT "LEFT$(A$,5) = ";LEFT$(A$,5)
70 PRINT
80 PRINT "RIGHT$(A$,3) = ";RIGHT$(A$,3)
90 PRINT
100 PRINT "RIGHT$(A$,5) = ";RIGHT$(A$,5)

110 PRINT
120 PRINT "MID$(A$,3) = ";MID$(A$,3)
130 PRINT
140 PRINT "MID$(A$,5) = ";MID$(A$,5)
150 PRINT
160 PRINT "MID$(A$,5,4) = ";MID$(A$,5,4)

170 PRINT
180 PRINT "MID$(A$,2,7) = ";MID$(A$,2,7)
```

Come puoi vedere, per prima cosa il programma pone A$ uguale a "QUINTA*STRADA". Poi usa LEFT$, RIGHT$ e MID$ per manipolare la stringa originale A$.

Ecco cosa accade quando si fa girare il programma:

```
LEFT$(A$,3) = QUI

LEFT$(A$,5) = QUINT

RIGHT$(A$,3) = ADA

RIGHT$(A$,5) = TRADA

MID$(A$,3) = INTA*STRADA

MID$(A$,5) = TA*STRADA

MID$(A$,5,4) = TA*S

MID$(A$,2,7) = UINTA*S
```

Guarda la prima riga del programma, LEFT$(A$,3)=QUI. LEFT$ prende tanti caratteri partendo dalla parte sinistra della stringa quanti sono indicati dal numero che segue la stringa. Cioe' quando abbiamo LEFT$(A$,3), esso prende i primi tre caratteri di sinistra della stringa. La prossima riga del programma, LEFT$(A$,5), prende i primi cinque caratteri di sinistra della stringa, producendo in questo caso QUINT.

LEFT$, MID$ e RIGHT$ possono essere usati piu' o meno nello stesso modo. Se abbiamo detto:

```
PRINT LEFT$("QUINTA*STRADA",3)
```

il computer stampera' QUI. La stringa puo' essere anche una stringa variabile (A$) o la stringa completa ("QUINTA*STRADA").

Come probabilmente immaginerai, RIGHT$ si comporta nella stessa maniera salvo che parte dalla destra della stringa. Percio' RIGHT$(A$,3) prende i primi tre caratteri di destra della stringa, in questo caso ADA. Ancora come sopra, e' lo stesso che dire:

```
PRINT RIGHT$("QUINTA*STRADA",3)
```

MID$ seleziona una parte all'interno della stringa partendo dal carattere posto nella posizione del numero che viene indicato dopo la stringa. Percio', MID$(A$,4) stampa tutta la stringa partendo dal quarto carattere.

Se c'e' un solo numero (come il quattro sopra), allora MID$ seleziona tutta la stringa fino alla fine. Comunque, se c'e' un altro numero, questo secondo numero detta la lunghezza della stringa che deve essere estratta.

Puoi vedere nelle ultime due righe del programma che MID$(A$,5,4) stama l'estratto della stringa lungo quattro caratteri partendo dal quinto carattere. MID$(A$,2,7) produce una stringa lunga sette caratteri, partendo dal secondo carattere.

Fai girare di nuovo il programma, inserendo il tuo nome al posto di QUINTA*STRADA in riga 20.

CONCATENAZIONE DI STRINGHE

Le stringhe possono essere sommate insieme sul COMMODORE 64. Questo processo prende il nome di concatenazione. Puoi concatenare due o piu' stringhe complete insieme, oppure aggiungere parti di loro come mostra il nostro prossimo programma:

```
10 REM CONCATENAZIONE DI STRINGHE
20 PRINT "J":REM CANCELLA LO SCHERMO
30 A$="AMERICA"
40 B$="COLOMBO"
50 C$=A$+B$
60 PRINT "A$ = ";A$
70 PRINT
80 PRINT "B$ = ";B$
90 PRINT
100 PRINT "C$ = ";C$
110 PRINT
120 D=INT(6*RND(1))+1
130 E=INT(6*RND(1))+7
140 PRINT "MID$(C$";D;",";E;") = ";MID$(
C$,D,E)
150 PRINT
```

```
160 D$=MID$(C$,D,E)
170 E$=A$+D$
180 PRINT "E$ = ";E$
```

Quando fai girare questo programma, che crea C$ in riga 100, concatenando A$ e B$, vedrai questo risultato:

```
A$ = AMERICA

B$ = COLOMBO

C$ = AMERICACOLOMBO

MID$(C$ 5 , 10 ) = ICACOLOMBO

E$ = AMERICAICACOLOMBO
```

NOME A PIRAMIDE

Si possono fare tante cose, manipolando le stringhe. Il NOME A PIRAMIDE ti permette di inserire il tuo nome producendo un programma molto interessante. Una volta fatto girare il programma capirai perche' gli e' stato dato questo nome.

Questo e' il listato:

```
10 REM NOME A PIRAMIDE
20 PRINT "♥":REM CANCELLA LO SCHERMO
30 INPUT "QUAL'E' IL NOME CHE VUOI INSER
IRE";A$
40 IF LEN(A$)>15 THEN A$=LEFT(A$,15)
50 A=LEN(A$)
60 PRINT "♥":REM CANCELLA LO SCHERMO
70 FOR G=1 TO A
80 PRINT TAB(18-G);
90 FOR H=1 TO 2*G
100 PRINT MID$(A$,G,1);
110 NEXT H
120 PRINT
130 NEXT G
```

Ecco due sviluppi del programma, uno adopera il nome di mio figlio, l'altro il nome di un personaggio che sicuramente avrai gia' sentito nominare:

```
          AA
         DDDD
        AAAAAA
       MMMMMMMM

     YYYYYYYYYYYY
    OOOOOOOOOOOOOO
   UUUUUUUUUUUUUUUU
  NNNNNNNNNNNNNNNNNN
 GGGGGGGGGGGGGGGGGGGG

          GG
         IIII
        UUUUUU
       LLLLLLLL
      IIIIIIIIII
     OOOOOOOOOOOO

     CCCCCCCCCCCCCCCC
    EEEEEEEEEEEEEEEEEE
   SSSSSSSSSSSSSSSSSSSS
  AAAAAAAAAAAAAAAAAAAAAA
 RRRRRRRRRRRRRRRRRRRRRRRR
EEEEEEEEEEEEEEEEEEEEEEEEEE
```

IL GIOCO DELL' ECO

Il nostro ultimo programma per questo capitolo, mostra un uso effettivo della manipolazione di stringhe, in cui una stringa e' progressivamente ridotta di un elemento.

Quando fai girare IL GIOCO DELL' ECO, vedrai una lettera apparire sullo schermo; dopo poco sparira'. Una volta sparita, ti sara' dato un tempo limitato per premere il tasto corrispondente.

Se hai premuto il tasto giusto, la lettera sara' sostituita con una nuova e si ripetera' il gioco. Ogni

volta che appare una nuova lettera, ti verra' dato sempre meno tempo per vederla, prima di schiacciare il tasto corrispondente sulla tastiera. Se sbagli, il "SPIACENTE QUESTO E' SBAGLIATO" apparira' insieme al punteggio da te ottenuto. Se riesci a inserire tutte le lettere giuste, ti sara' dato il messaggio "SEI IL CAMPIONE".

Ecco il listato:

```
10 REM GIOCO DELL'ECO
20 REM IL PUNTEGGIO MASSIMO E' 30
30 S=0
40 A$="ZXACSDVFBGHTYREWQNHFYUJMPKOLEU"
50 PRINT "□":REM CANCELLA LO SCHERMO
60 PRINT:PRINT:PRINT:PRINT:PRINT
70 PRINT TAB(19) MID$(A$,1,1)
80 FOR G=1 TO 10*LEN(A$):NEXT G
90 PRINT "□":REM CANCELLA LO SCHERMO
100 GET B$
110 IF B$<"A" OR B$>"Z" THEN C=C+1:IF C<
500 GOTO 100
120 PRINT:PRINT:PRINT:PRINT
130 PRINT TAB(19) B$
140 IF B$=MID$(A$,1,1) THEN S=S+1
150 PRINT:PRINT:PRINT TAB(9) "IL TUO PUN
TEGGIO E'";S
190 IF B$<>MID$(A$,1,1) THEN GOTO 240
200 A$=MID$(A$,2)
210 IF LEN(A$)=1 THEN GOTO 270
220 FOR G=1 TO 2000:NEXT G
230 GOTO 60
240 PRINT:PRINT:PRINT TAB(7) "SPIACENTE
QUESTO E' SBAGLIATO"
250 PRINT:PRINT:PRINT TAB(12) "HAI OTTEN
UTO";S
260 END
270 PRINT:PRINT:PRINT TAB(12) "SEI IL CA
MPIONE"
```

La variabile S, che tiene il tuo punteggio, e' posta uguale a zero in riga 30, e la riga 40 pone la variabile di stringa A$ uguale ad una lunga fila di lettere. La riga 70 stampa la prima lettera della stringa, e la riga 80 inserisce una breve pausa usando la funzione LEN.

Questa e' un'altra funzione di stringa e tratta la lunghezza della stringa, cioe' il numero di caratteri che contiene. LEN non fa nessuna distinzione tra lettere, numeri, simboli o spazi come puoi vedere inserendo un certo numero di informazioni PRINT LEN (A$), dopo aver posto A$ uguale a delle parole, simboli e frasi.

Dato che, nel nostro programma, A$ e' ridotto ad un carattere (riga 200) ogni volta che il programma compie un giro, LEN (A$) diventera' un numero sempre piu' piccolo. Percio', la pausa prodotta in riga 80 (che detta per quanto tempo la lettera rimane sullo schermo prima di scomparire) diventera' piu' corta.

GET

La riga 100 usa GET per leggere dalla tastiera. GET, come ti sarai gia' accorto a questo punto del libro, non richiede che tu prema RETURN dopo aver toccato un tasto. GET considera sempre il tasto premuto come una stringa. Al contrario di INPUT non aspetta che tu abbia premuto un tasto prima di continuare il programma.

Se non stai toccando un tasto quando il programma arriva a GET, esso passa oltre considerando il tuo non toccare la tastiera, come una stringa nulla (due virgolette con niente, nemmeno uno spazio, tra di loro, come "").

Se vuoi fermare il tuo programma ad un certo punto per aspettare un input dalla tastiera, ti sara' necessario usare una riga come questa:

```
100 GET A$:IF A$="" THEN GOTO 100
```

Il programma si manterra' su questa riga fino a che non si prema un tasto. La riga 110 considera la variabile B$, posta uguale a qualsiasi tasto si schiacci. Come puoi vedere, puoi usare i simboli maggiore di (>) e minore di (<), che abbiamo visto nel capitolo precedente, in

connessione con le stringhe. Questi guardano tutti gli elementi di una stringa e li paragonano in termini di ordine alfabetico (cosi' ZEBRA e' minore di ACQUA e GRANDE e' maggiore GIUSTO).

Nello stesso modo puoi paragonare stringhe usando l' uguale (=) e il diverso da (<>) come mostrato nelle righe successive del programma. La riga 140 paragona il tasto da te premuto con il primo elemento della A$, e, se coincidono, continua ad aggiungere uno al tuo punteggio (variabile S).

La riga 190 paragona B$ con il primo elemento della A$, e, se non coincidono, manda il programma alle righe 240 e 250 dove ti si dice "SPIACENTE QUESTO E' SBAGLIATO" e ti si da' il punteggio ottenuto.

La riga 200 toglie alla stringa A$ il suo primo carattere, ponendo A$ uguale a MID$(A$,2). La riga 210 controlla se la lunghezza di A$ e' uguale ad uno (cioe' se LEN A$=1) e se cio' e' verificato, va alla riga 270 e stampa il messaggio "TU SEI IL CAMPIONE". Altrimenti, il programma torna alla riga 60 per stampare la prossima lettera.

CAPITOLO 11

I COMANDI READ DATA E RESTORE

In questo capitolo, esamineremo tre parole del nostro vocabolario: READ, DATA e RESTORE. Esse sono usate per portare informazioni immagazzinate in una parte del programma ad un'altra parte. Inserisci e fai girare questo programma che ti chiarira' la funzione:

```
10 REM READ,DATA E RESTORE
20 DIM A(5)
30 FOR B=1 TO 5
40 READ A(B)
50 PRINT A(B)
60 NEXT B
70 DATA 88,8965,23,-94,3
```

Usando la riga 40, il programma legge le informazioni DATA in riga 70, seguendo l'ordine, stampando ciascuno dei DATA mediante la riga 50.

RESTORE riporta il COMMODORE 64 al primo dei DATA nel programma, come potrai scoprire se modificherai il programma suddetto aggiungendo la riga 55; cosi' risultera' il seguente listato:

```
10 REM READ,DATA E RESTORE
20 DIM A(5)
30 FOR B=1 TO 5
40 READ A(B)
50 PRINT A(B)
55 IF B=3 THEN RESTORE
60 NEXT B
70 DATA 88,8965,23,-94,3
```

Non importa dove sono immagazzinati i DATA nel programma. Il computer li trovera', in ordine dal primo dei DATA del programma fino all'ultimo, come il nostro seguente programma dimostra:

```
1 DATA 45
10 REM READ,DATA E RESTORE
20 DIM A(5)
22 DATA 888
30 FOR B=1 TO 5
40 READ A(B)
50 PRINT A(B)
55 DATA 432
60 NEXT B
70 DATA 933,254
```

READ e DATA si comportano nello stesso modo con le informazioni di stringa:

```
10 REM READ/DATA CON STRINGHE
20 FOR B=1 TO 5
30 READ A$
40 PRINT A$
50 NEXT B
60 DATA TEST,UNO,NOVE,DOPO,GIORNO
```

Nota che le stringhe dei DATA non devono essere racchiusi tra virgolette, e/o la punteggiatura e i simboli sono significanti e devono essere considerati parti dei DATA.

Puoi mischiare DATA numerici e DATA di stringhe nello stesso programma, ma assicurati che quando il programma vuole un dato numerico, un numero arrivi nel programma, e quando vuole una stringa, la trovi:

```
10 REM READ/DATA CON STRINGHE
20 FOR B=1 TO 5
30 READ A$:READ A
40 PRINT A$:PRINT A
50 NEXT B
60 DATA TEST,12,UNO,989892,NOVE,3,DOPO,-
892781,GIORNO,23
```

C A P I T O L O 12

C R E I A M O L A M U S I C A

Il COMMODORE 64 e' provvisto di un sintetizzatore a tre voci molto sofisticato.Tre voci significa che puo' suonare tre note o toni nello stesso tempo. Comunque, per semplificare, e perche' sia piu' facile sentire i risultati dell'esperimento che stiamo per fare, useremo una sola voce.

Il COMMODORE 64 e' capace di produrre una scala di otto ottave, possiede un controllo di volume regolabile, quattro differenti forme d'onda per scegliere il suono, attack/decay e sustain/release regolabili e una regolabile media d'incremento del rettangolare.

Tutte queste caratteristiche sono attivate o modificate dai valori POKE nell'indirizzo di memoria corrispondente. Ricorderai che abbiamo discusso di POKE e PEEK nel capitolo sui grafici. Cio' che stiamo per fare nel seguente capitolo e' essenzialmente lo stesso, ma il risultato sara' in suoni invece che in forme colorate sullo schermo. Le locazioni POKE che stiamo per usare sono per una voce sola. Una lista delle locazioni POKE per le tre voci con una tavola di tutte le note musicali e i loro valori corrispondenti appare alla fine di questo capitolo per un riferimento piu' immediato.

VOLUME POKE 54296,n

Il volume deve essere acceso prima di produrre il suono. Una volta acceso il volume all'inizio di un programma sonoro, non sara' piu' necessario ripetere l'operazione. Il valore che puo' essere POKato varia da 1 a 15. Zero spegne il volume e 15 e' il volume massimo ché il computer puo' produrre. L'indirizzo POKE per il volume e' lo stesso per tutte le tre voci.

FORMA D' ONDA POKE 54276,n

La forma d'onda influenza il carattere sonoro di una tonalita'. Il computer puo' produrre quattro differenti forme d'onda, triangolare, dente di sega, rumore, rettangolare. Tratteremo di queste un po' piu' tardi e scopriremo le loro differenze. Queste forme d'onda possono essere manipolate per riprodurre vari strumenti. Per esempio, la forma d'onda triangolare puo' riprodurre il suono di un flauto o di uno xilofono, la forma d'onda dente di sega, un'arpa e quella rettangolare, un pianoforte. Il rumore e' molto utile per generare suoni ad effetti diversi. Comunque, generalmente sono molto variabili.
Il numero che segue l'indirizzo POKE deve essere:

TRIANGOLARE	17
DENTE DI SEGA	33
RETTANGOLARE	65
RUMORE	129

Questi numeri sono gli stessi per tutte le tre voci. Nota: se usi POKE 54276,65 per avere la forma d'onda rettangolare avrai anche bisogno di una media di incremento. Questo consiste in una alta e una bassa media di incremento, usata per controllare il suono della nota.

Alta media di incremento POKE 54275,n
Bassa media di incremento POKE 54274,n

Per l'alta media il valore puo' variare tra zero e 15, per quella bassa il valore puo' variare tra zero e 255. Queste medie di incremento possono essere usate solo con le forme d'onda rettangolari.

ATTACK/DECAY POKE 54277,n

Il sistema ATTACK/DECAY controlla la modulazione di una tonalita' fino a che la nota arriva al suo punto piu' alto di volume (ATTACK) e quando cade dal suo livello di volume piu' alto (DECAY). Il valore POKato in questo sistema e' ottenuto dalla seguente lista. Per ottenere il corretto

valore cerca il valore della media di incremento di ATTACK che desideri usare e il valore di DECAY e addizionali insieme. Il totale di questi due numeri e' il valore POKato nei registri prestabiliti. Ad esempio un'alta media di incremento di ATTACK con un valore di 128 aggiunta ad un'alta media di decremento di DECAY con un valore di 8 dara' un totale di 136. Cosi' si potra' POKare 54277,136.

alto ATTACK	medio ATTACK	basso ATTACK	bassissimo ATTACK
128	64	32	16

alto DECAY	medio DECAY	basso DECAY	bassissimo DECAY
8	4	2	1

Le medie di incremento di ATTACK possono essere ancor piu' incrementate aggiungendo due o piu' altri ATTACK. Se aggiungi per esempio un alto ATTACK ad un medio ATTACK, avrai un valore di 192 e poi aggiungi un medio DECAY e avrai un valore POKE di 196.

I valori ATTACK/DECAY sono gli stessi per ogni voce.

SUSTAIN/RELEASE POKE 54278,n

Il sistema SUSTAIN/RELEASE e' opzionale. Il COMMODORE 64 puo' emettere la nota anche senza di esso, ma esso vi dara' il controllo della nota molto piu' precisamente.

Il sistema SUSTAIN/RELEASE si combina con l'andamento ATTACK/DECAY. Esso ti permette di mantenere una nota al suo livello di volume piu' alto prima che questo inizi a diminuire. Il sistema SUSTAIN/RELEASE puo' anche essere inserito in un ciclo FOR/NEXT .

alto SUSTAIN	medio SUSTAIN	basso SUSTAIN	bassissimo SUSTAIN
128	64	32	16

alto RELEASE	medio RELEASE	basso RELEASE	bassissimo RELEASE
8	4	2	1

Il sistema SUSTAIN/RELEASE e' usato nello stesso modo di ATTACK DECAY. Si possono aggiungere due o piu' valori insieme per avere il valore POKE. Questi valori sono gli stessi per tutte le tre voci.

L' EMISSIONE DI UNA NOTA

Ogni nota musicale, in tutte le otto ottave, ha due valori: alta frequenza e bassa frequenza. Il valore di ognuno di questi, per tutte le note disponibili, si puo' trovare nella lista alla fine di questo capitolo. Il computer combina questi due valori per produrre la nota.

L'ultima cosa di cui abbiamo bisogno prima di poter produrre una nota, e' di specificare la lunghezza della nota; per fare cio' useremo un ciclo FOR/NEXT come una pausa. Un ciclo di 250 corrisponde a una nota di un quarto, 500 una nota di un mezzo, 125 una nota di un ottavo e cosi' via. Bisogna notare che queste pause sono approssimate e che potrebbero essere modificate dalla lunghezza del programma.

Prima di provare ad emettere un motivo, iniziamo con un breve programma che emette una sola nota e usiamolo per esplorare le varie sezioni. Inserisci il seguente programma; puoi tralasciare le istruzioni REM, se vuoi, io le ho inserite in modo tale che tu possa vedere la funzione di ogni riga. NOTA: i sistemi ATTACK/DECAY e SUSTAIN/RELEASE devono essere definiti prima della forma dell'onda.

```
10 POKE 54296,15:REM VOLUME
20 POKE 54277,130:REM ATTACK/DECAY
30 POKE 54278,130:REM SUSTAIN/RELEASE
40 POKE 54273,34:POKE 54272,75:REM VALOR
E DELLA NOTA DO NELL'OTTAVA 5
50 POKE 54276,17:REM FORMA DELL'ONDA
60 FOR D=1 TO 500:NEXT:REM RITARDO PER L
A DURATA DELLA NOTA
70 POKE 54276,0:POKE 54277,0:POKE 54278,
0
80 REM FORMA D'ONDA,ATTACK/DECAY,SUSTAIN
/RELEASE DEVONO ESSERE SPENTI DOPO L'USO
```

Dopo aver fatto girare questo programma per alcune volte in modo da sentire la nota prodotta, inizia a cambiare alcune istruzioni. Prima di tutto cambia la forma dell'onda in riga 50 per sentire la differenza tra i quattro differenti tipi. Ricordati che quando hai una forma dell'onda rettangolare (65), dovrai aggiungere una riga extra per provvedere alla sua media di incremento. Inizia con questa riga e poi procedi con i cambi.

```
55 POKE 54275,0:POKE 54274,255
```

Ora cambia i valori nei sistemi ATTACK/DECAY e SUSTAIN/RELEASE e osserva la versatilita' del tuo COMMODORE 64. Cambia le riga 20, 30 e 60:

```
20 POKE 54277,17:REM ATTACK/DECAY
30 POKE 54278,17:REM SUSTAIN/RELEASE
60 FOR D=1 TO 100:NEXT:REM RITARDO PER L
A DURATA DELLA NOTA
```

Quando sei pronto prova questa nuova versione del programma:

```
5 FOR D=15 TO 1 STEP-1
10 POKE 54296,D:REM VOLUME
20 POKE 54277,17:REM ATTACK/DECAY
30 POKE 54278,196:REM SUSTAIN/RELEASE
40 POKE 54273,34:POKE 54272,75:REM VALOR
E PER LA NOTA DO NELL' OTTAVA 5
50 POKE 54276,65:REM FORMA DELL'ONDA
55 POKE 54275,0:POKE 54274,255
60 NEXT
70 POKE 54276,0:POKE 54277,0:POKE 54278,
0
80 REM FORMA D'ONDA,ATTACK/DECAY,SUSTAIN
/RELEASE DEVONO ESSERE SPENTI DOPO L'USO
```

In questa versione abbiamo iniziato la pausa dalla riga 5 e abbiamo fatto in modo che il ciclo FOR/NEXT andasse all'indietro, partendo da 15. Abbiamo poi usato questa

variabile per il valore POKE che controlla il volume. Per far cio' cambiamo il livello del volume ogni volta che il programma attraversa il ciclo, cosi' il volume della nota si affievolira' sempre piu'.

Altri esperimenti fatti da parte tua ti mostreranno la versatilita' del sintetizzatore del tuo COMMODORE 64.

Prima di finire questa sezione, ecco un programma che suonera' una parte di una melodia ben conosciuta, usando le istruzioni READ e DATA. Inserisci e fai girare il programma e poi ritorna sulla spiegazione del listato.

```
10 V=54296:A=54277:S=54278:W=54276
20 HP=54275:LP=54274
30 HF=54273:LF=54272
40 POKE V,15:POKE A,9:POKE S,9
50 POKE W,65:POKE HP,1:POKE LP,250
60 READ H
70 READ L
80 READ D
90 IF H=999 THEN POKE W,0:POKE A,0:POKE
S,0:END
100 POKE HF,H:POKE LF,L
110 FOR T=1 TO D:NEXT:POKE HF,0:POKE LF,
0:POKE W,0
120 GOTO 40
130 DATA 21,154,250,25,177,500,28,214,25
0,32,94,325,34,75,125,32,94,250
140 DATA 28,214,500,22,227,250,19,63,325
,21,154,125,22,227,250
150 DATA 25,177,500,21,154,250,21,154,32
5,20,100,125,21,154,250,22,227,500
160 DATA 19,63,250,16,47,500,21,154,250
170 DATA 25,177,500,28,214,250,32,94,325
,34,75,125,32,94,250
180 DATA 28,214,500,22,227,250,19,63,325
,21,154,125,22,227,250
190 DATA 25,177,325,22,227,125,21,154,25
0,19,63,325,18,42,125
200 DATA 19,63,250,21,154,500,21,154,250
,21,154,750
1000 DATA 999,999,999
```

Le righe 10, 20 e 30 assegnano differenti nomi alle locazioni POKE; cio' elimina la necessita' di stampare la locazione ogni volta che questa viene usata. La riga 40 definisce il volume e gli andamenti di ATTACK/DECAY e SUSTAIN/RELEASE. La riga 50 definisce la forma d'onda e le alte e basse medie di incremento del rettangolare.

Leggiamo (READ) poi il valore per l'alta frequenza, H, per la bassa frequenza, L, e per il valore di durata, D, che e' usato nel ciclo FOR/NEXT in riga 110 per determinare la durata della nota.

La riga 1000 contiene i valori che determinano la fine del programma (riga 90). Quando sono stati letti tutti i relativi dati, allora il prossimo valore trovato in riga 60 e' 999. Questo non e' un valore legale per una nota cosi' lo usiamo per dire al computer che il motivo e' finito e di spegnere la forma dell'onda, e le locazioni ATTACK/DECAY e SUSTAIN/RELEASE.

La riga 100 prende i valori correnti di H e di L e li usa per emettere la nota desiderata. La riga 110 fornisce la corretta lunghezza della nota. Inoltre, si annullano i valori di alta e bassa frequenza, preparandoli per il prossimo giro. Ogni ripetizione del ciclo e' considerata un programma a se' stante. La riga 120 rimanda il programma all'inizio, per far si' che venga emessa la nota successiva.

Le istruzioni DATA sono ordinate in modo tale che le informazioni per ogni nota si trovino nell'ordine corretto per le istruzione READ. Cioe', i valori di alta frequenza, bassa frequenza e durata della nota.

VOLUME POKE 53296 (uguale per tutte le tre voci)

VOCE 1

ALTA FREQUENZA POKE 54273 BASSA FREQUENZA POKE 54272 FORMA DELL' ONDA POKE 54276
ALTO RETTANGOLARE POKE 54275 BASSO RETTANGOLARE POKE 54274 (solo con forma d'onda rettangolare)
ATTACK/DECAY POKE 54277
SUSTAIN/RELEASE POKE 54278

VOCE 2

ALTA FREQUENZA POKE 54280 BASSA FREQUENZA POKE 54279 FORMA DELL' ONDA POKE 54288
ALTO RETTANGOLARE POKE 54282 BASSO RETTANGOLARE POKE 54281
(solo con forma d'onda rettangolare)
ATTACK/DECAY POKE 54284
SUSTAIN/RELEASE POKE 54286

VOCE 3

ALTA FREQUENZA POKE 54287 BASSA FREQUENZA POKE 54286 FORMA DELL' ONDA POKE 54290
ALTO RETTANGOLARE POKE 54289 BASSO RETTANGOLARE POKE 54288
(solo con forma d'onda rettangolare)
ATTACK/DECAY POKE 54291
SUSTAIN/RELEASE POKE 54292

OTTAVA	NOTA	ALTA FREQUENZA	BASSA FREQUENZA
2	RE#	5	25
2	MI	5	103
2	FA	5	185
2	FA#	6	16
2	SOL	6	108
2	SOL#	6	206
2	LA	7	53
2	LA#	7	163
2	SI	8	23
3	DO	8	147
3	DO#	9	21
3	RE	9	159
3	RE#	10	60
3	MI	10	205
3	FA	11	114
3	FA#	12	32
3	SOL	12	216
3	SOL#	13	156
3	LA	14	107

OTTAVA	NOTA	ALTA FREQUENZA	BASSA FREQUENZA
3	LA#	15	70
3	SI	16	47
4	DO	17	37
4	DO#	18	42
4	RE	19	63
4	RE#	20	100
4	MI	21	154
4	FA	22	227
4	FA#	24	63
4	SOL	25	177
4	SOL#	27	56
0	DO	1	18
0	DO#	1	35
0	RE	1	52
0	RE#	1	70
0	MI	1	90
0	FA	1	110
0	FA#	1	132
0	SOL	1	155
0	SOL#	1	179
0	LA	1	205
0	LA#	1	233
0	SI	2	6
1	DO	2	37
1	DO#	2	69
1	RE	2	104
1	RE#	2	140
1	MI	2	179
1	FA	2	220
1	FA#	3	8
1	SOL	3	54
1	SOL#	3	103
1	LA	3	155
1	LA#	3	210
1	SI	4	12
2	DO	4	73
2	DO#	4	139
2	RE	4	208

OTTAVA	NOTA	ALTA FREQUENZA	BASSA FREQUENZA
4	LA	28	214
4	LA#	30	141
4	SI	32	94
5	DO	34	75
5	DO#	36	85
5	RE	38	126
5	RE#	40	200
5	MI	43	52
5	FA	45	198
5	FA#	48	127
5	SOL	51	97
5	SOL#	54	111
5	LA	57	172
5	LA#	61	126
5	SI	64	188
6	DO	68	149
6	DO#	72	169
6	RE	76	252
6	RE#	81	161
6	MI	86	105
6	FA	91	140
6	FA#	96	254
6	SOL	102	194
6	SOL#	108	223
6	LA	115	88
6	LA#	122	52
6	SI	129	120
7	DO	137	43
7	DO#	145	83
7	RE	153	247
7	RE#	163	31
7	MI	172	210
7	FA	183	25
7	FA#	193	252
7	SOL	205	133
7	SOL#	217	189
7	LA	230	176
7	LA#	244	103

CAPITOLO 13

GLI ARRAYS

Un array viene usato quando si vuole usare una lista di oggetti, e far riferimento ad uno di questi notificando la sua posizione all'interno della lista. Si dimensiona un array usando il comando DIM. Se scrivi DIM A(20), il COMMODORE 64 definira' nella sua memoria una lista chiamata A, cioe' uno spazio di 21 oggetti: A(0), A(1), e cosi' via fino ad A(20). Ciascuno di questi oggetti viene chiamato 'elemento dell'array'.

Quando si dimensiona un array, il computer crea una lista nella sua memoria e inserisce ogni oggetto in questa lista con uno zero. Cosi', se dici al computer di stampare PRINT A(3), esso stampera' uno zero. Si dovranno assegnare dei valori agli elementi di un array con un'informazione come A(2)=1000, o usando READ e DATA come abbiamo visto nel capitolo 11. Una volta dato un valore a un elemento, puoi fare in modo che il computer ti dia questo valore, dicendogli PRINT A(n). Puoi anche manipolare l'elemento come se fosse un numero. Cioe', A(4)*6 e' valido, cosi' come 45-A(6) e cosi' via.

Il COMMODORE 64 ti consente di usare un array fino ad 11 elementi (cioe' da A(0) fino ad A(10)), senza dover usare la istruzione DIM all'inizio. Quando incontra un riferimento ad un elemento dell'array, dove il 'subscript' (il numero che segue nella parentesi) e' compreso tra zero e dieci, il computer automaticamente crea un array. Comunque, e' bene dimensionare sempre gli arrays, anche se stai usando meno di 12 elementi.

Potresti anche dimenticare gli elementi che hanno come subscript zero e pretendere che l'array parta da uno. Molte volte scoprirai che e' piu' semplice pensare che DIM A(80) ti dia un array di 80 elementi (piuttosto che 81 come e' in realta'), e che il primo elemento sia A(1).

Il primo programma in questo capitolo dimensiona un array

chiamato A, con spazio per sedici elementi. Ignoreremo l'elemento con il subscript zero. Il ciclo B, da riga 40 a riga 60, genera un array con cifre a caso tra zero e nove, e poi le stampa con il ciclo da riga 70 a riga 100 (con una piccola pausa in riga 90).

Ecco il listato:

```
10 REM ARRAYS
20 DIM A(15)
30 PRINT "♥":REM CANCELLA LO SCHERMO
40 FOR B=1 TO 15
50 A(B)=INT(9*RND(1))
60 NEXT B
70 FOR Z=1 TO 15
80 PRINT "A(";Z;") E'";A(Z)
90 FOR T=1 TO 500:NEXT T
100 NEXT Z
```

Ed ecco il programma:

```
A( 1 ) E' 3
A( 2 ) E' 5
A( 3 ) E' 2
A( 4 ) E' 4
A( 5 ) E' 5
A( 6 ) E' 5
A( 7 ) E' 6
A( 8 ) E' 2
A( 9 ) E' 3
A( 10 ) E' 3
A( 11 ) E' 1
A( 12 ) E' 1
A( 13 ) E' 7
A( 14 ) E' 0
A( 15 ) E' 7
```

Questo si chiama array unidimensionale, perche' una sola cifra segue la lettera o il nome che definisce l'array.

Si possono avere anche array multidimensionali, in cui piu' di un numero segue la definizione dell'array dopo DIM.

Nel nostro prossimo programma, per esempio, il computer genera un array bidimensionale chiamato ancora A, di cinque elementi per cinque (cioe', e' dimensionato da DIM A(4,4) come puoi vedere in riga 20):

```
10 REM ARRAYS MULTI-DIMENSIONALI
15 POKE 53281,14
20 DIM A(4,4)
30 PRINT "♥":REM CANCELLA LO SCHERMO
40 FOR B=1 TO 4
50 FOR C=1 TO 4
60 A(B,C)=INT(9*RND(1))
70 NEXT C
80 NEXT B
90 PRINT:PRINT:PRINT:PRINT:PRINT
100 PRINT "■    1  2  3  4":REM COLORE N
ERO
110 FOR B=1 TO 4
120 PRINT "■" B;:REM COLORE NERO
130 FOR C=1 TO 4
140 PRINT "■" A(B,C);:REM COLORE BIANCO
150 NEXT C
160 PRINT
170 NEXT B
```

Ecco cosa appare quando fai girare il programma:

```
     1  2  3  4
  1  5  1  4  1
  2  4  5  1  0
  3  6  6  1  7
  4  6  8  7  4
```

Si puo' specificare l'elemento dell'array bidimensionale riferendosi a entrambi i suoi numeri, cosi' che l'elemento 1,1 di questo array (l'elemento all'angolo superiore sinistro del programma) e' cinque ed e' conosciuto come A(1,1). L'otto del programma e' A(4,2), e il sei sopra di esso e' A(3,2).

Il tuo computer puo' avere anche arrays di stringhe. Inserisci e fai girare questo breve programma per vedere in opera gli arrays di stringa:

```
10 REM ARRAYS DI STRINGHE
20 DIM A$(5)
30 PRINT "♥":REM CANCELLA LO SCHERMO
40 FOR B=1 TO 5
50 A$(B)=CHR$(INT(26*RND(1))+65)+CHR$(IN
T(26*RND(1))+65)
60 NEXT B
70 FOR B=1 TO 5
80 PRINT "A$(";B;") E' ";A$(B)
90 NEXT B
```

Ecco il 'printout' del programma:

```
A$( 1 ) E' GX
A$( 2 ) E' LN
A$( 3 ) E' ZP
A$( 4 ) E' BT
A$( 5 ) E' NP
```

Puoi naturalmente immettere gli elementi di un array, numerici o di stringa, mediante istruzioni DATA o INPUT. Ecco un array di stringa che viene formato leggendo le istruzioni DATA:

```
10 REM ARRAYS DI STRINGHE
20 DIM A$(5)
30 PRINT "♥":REM CANCELLA LO SCHERMO
40 FOR B=1 TO 5
50 READ A$(B)
60 NEXT B
70 FOR B=1 TO 5
80 PRINT "A$(";B;") E' ";A$(B)
90 NEXT B
100 DATA PENSARE, E', UN, FATICOSO, DOVE
RE
```

Ecco il risultato:

```
A$( 1 ) E' PENSARE
A$( 2 ) E' E'
A$( 3 ) E' UN
A$( 4 ) E' FATICOSO
A$( 5 ) E' DOVERE
```

IL GIOCO DELLA GABBIA

Il prossimo programma mostra l'uso di un array bidimensionale per imprigionare un oggetto e farlo muovere nell'array. L'oggetto in questo caso e' un cerchietto (drago). Questi e' intrappolato in una gabbia oscura, e muovendosi totalmente a caso, spera un giorno di uscirne. Sara' libero solo quando riuscira' a capitare sulle sbarre esterne.

Ecco il listato del programma:

```
10 REM LA TANA DEL DRAGO
30 DIM A(10,10)
40 POKE 53280,0:POKE 53281,11
50 PRINT "♥":REM CANCELLA LO SCHERMO
60 M=0
70 GOSUB 300
80 IF RND(1)> .35 THEN P=P+1:GOTO 100
90 P=P-1
100 IF RND(1)> .35 THEN Q=Q+1:GOTO 110
105 Q=Q-1
110 IF Q<1 THEN Q=1
120 IF Q>10 THEN Q=10
130 IF P<1 THEN P=1
140 IF P>10 THEN P=10
150 M=M+1
155 PRINT "■":REM HOME CURSOR
160 PRINT "■■■";:REM CURSORE GIU' UTILIZ
ZATO * 3 VOLTE
170 PRINT TAB(12) "■ NUMERO A CASO"
180 A(P,Q)=113
190 PRINT "■■■";:REM CURSORE GIU' UTILIZ
ZATO * 3 VOLTE
```

```
200 FOR X=1 TO 10
210 FOR Y=1 TO 10
220 PRINT TAB(10) "■" CHR$(A(X,Y));" ";:
REM BIANCO
230 NEXT Y
240 PRINT:PRINT TAB(6),
260 NEXT X
270 A(P,Q)=32
280 IF Q<9 OR P<9 THEN GOTO 80
290 GOTO 370
300 Q=INT(3*RND(1))+4
310 P=INT(3*RND(1))+4
320 FOR X=1 TO 10
330 FOR Y=1 TO 10
340 A(X,Y)=43
350 NEXT Y,X
360 RETURN
370 PRINT:PRINT
380 PRINT TAB(9) "■■ BENE...LIBERO FINAL
MENTE!■":REM REV ON AZZURRO REV OFF
390 FOR T=1 TO 4000:NEXT
400 RUN
```

C A P I T O L O 14

P R O G R A M M I D I R E L A X

Hai lavorato molto duramente ed e' percio' ora di divertirsi; questo capitolo conclusivo non sara' di studio, ma potrai giocare con e contro il tuo computer. I programmi non ti porteranno via molto tempo per l'inserimento, e sono veramente divertenti.

EXTRA-TERRESTRI

Nelle profondita' del tuo COMMODORE 64 si trova una misteriosa societa' di extra-terrestri. Sebbene questi siano creature gentili e semplici, hanno un elevato sistema sociale, regolato da un severo sistema di leggi. Queste leggi controllano la nascita, la crescita e la morte degli extra-terrestri da una generazione all'altra. Il comportamento di questi extra-terrestri fa in modo che queste creature si riuniscano in gruppi chiusi, viventi in griglie di 14 per 14. Percio' e' possibile, per un extra-terrestre trovantesi vicino al centro della griglia, avere fino a otto vicini. Una creatura sulle righe di confine puo' avere 5 vicini, e quella in un angolo della griglia, 3. La legge degli extra-terrestri dice che ogni creatura con due o tre vicini sopravvivera' fino alla successiva generazione. Un gruppo di tre creature da' vita a un nuovo alieno. Un'extra-terrestre con piu' di tre vicini morira' per mancanza di cibo. Anche una o due creature da sole moriranno, ma questa volta a causa della solitudine.

Queste regole sono rigide ma producono interessanti risultati.

Ecco il listato del programma che ti permette di osservare come queste creature vivano e muoiano all'interno del computer. La righe da 100 a 240 regolano le leggi della societa' degli alieni. Le righe da 1000 a 1090 stampano sullo schermo il risultato di ogni generazione. Dopo il

listato abbiamo inserito alcune parti del programma. Questo gioco e' basato sul gioco "LIFE" di John Conway.

```
5 REM EXTRA-TERRESTRI
7 PRINT "♥":REM CANCELLA LO SCHERMO
9 POKE 53280,0:POKE 53281,0
10 DIM A(14,14):DIM B(14,14)
20 FOR X=2 TO 13
30 FOR Y=2 TO 13
50 IF RND(1)>.5 THEN A(X,Y)=1
60 B(X,Y)=A(X,Y)
70 NEXT Y
80 NEXT X
85 G=1
90 GOSUB 1000
95 G=G+1
100 FOR X=2 TO 13
110 FOR Y=2 TO 13
120 C=0:T=0
130 IF A(X-1,Y-1)=1 THEN C=C+1
140 IF A(X-1,Y)=1 THEN C=C+1
150 IF A(X-1,Y+1)=1 THEN C=C+1
160 IF A(X,Y-1)=1 THEN C=C+1
170 IF A(X,Y+1)=1 THEN C=C+1
180 IF A(X+1,Y-1)=1 THEN C=C+1
190 IF A(X+1,Y)=1 THEN C=C+1
200 IF A(X+1,Y+1)=1 THEN C=C+1
210 IF A(X,Y)=1 AND C<>2 AND C<>3 THEN B
(X,Y)=0
220 IF A(X,Y)=0 AND C=3 THEN B(X,Y)=1
230 NEXT Y
240 NEXT X
250 GOTO 90
1000 PRINT
1001 GOSUB 1100
1002 PRINT "♥":REM HOME CURSOR
1003 PRINT TAB(10) "[illegible] GENERAZIONE >>>
";G:REM CURSORE GIU' PORPORA
1005 PRINT
1010 FOR X=1 TO 14
1020 FOR Y=1 TO 14
```

```
1030 A(X,Y)=B(X,Y)
1040 IF A(X,Y)=1 THEN PRINT TAB(12) "█"
CHR$(113);:T=T+1:REM GIALLO
1050 IF A(X,Y)=0 THEN PRINT TAB(12)
1060 NEXT Y
1070 PRINT
1080 NEXT X
1082 POKE 1932,32
1085 PRINT TAB(7) "███NUMERO DI EXTRA-TE
RRESTRI";T:REM CURSORE GIU' PORPORA
1090 RETURN
1100 POKE 54296,15:POKE 54277,24:POKE 54
278,136
1110 POKE 54273,25:POKE 54272,177
1120 POKE 54276,17
1130 FOR D=1 TO 200:NEXT
1140 POKE 54276,0:POKE 54277,0:POKE 5427
8,0
1150 RETURN
```

GENERAZIONE >>> 1

GENERAZIONE >>> 2
GENERAZIONE >>> 3
GENERAZIONE >>> 4
GENERAZIONE >>> 5
GENERAZIONE >>> 6
GENERAZIONE >>> 7

UN ALTRO GIOCO DI DADI

Il programma finale di questo capitolo e' un semplice gioco di dadi. Il programma usa due arrays (B e D), che abbiamo studiato nel capitolo precedente, per immagazzinare rispettivamente i punteggi dei giocatori e le gettate dei dadi.

E' un gioco facile. Siete in gara tu e il computer per raggiungere un punteggio totale di 50 o piu'. Si gettano due dadi alla volta, ma la sola volta che si ottiene un punteggio e' quando entrambi i dadi totalizzano lo stesso numero. Quando cio' si verifica, moltiplicherai tre volte il valore di uno dei dadi e lo aggiungerai al tuo precedente punteggio.

Dovresti ormai saperne abbastanza per essere capace di determinare la funzione di ogni sezione, cosi' non ti annoiero' spiegandoti questo programma relativamente semplice. Scrivi il listato e fai girare il programma. Probabilmente imparerai di piu' studiandotelo da solo che seguendo dettagliatamente le mie spiegazioni. Ecco il listato che conclude questo capitolo:

```
10 REM GIOCO DI DADI
20 POKE 53281,12:POKE 53280,7
40 DIM B(2):DIM D(2):R=1
45 FOR A=1 TO 2:PRINT "♥":REM CANCELLA L
O SCHERMO
50 PRINT:PRINT TAB(8) "■ >>> GIOCO DI DA
DI <<<":REM NERO
60 PRINT:PRINT TAB(14) "ROUND NO.";R
65 PRINT:PRINT
70 PRINT:PRINT TAB(14) "COMPUTER :";B(1)

80 PRINT:PRINT TAB(14) "UOMO    :  ";B(2)

90 IF B(1)>49 OR B(2)>49 THEN FOR DD=1 T
O 1500:NEXT:GOTO 400
100 IF A=1 THEN PRINT:PRINT:PRINT TAB(10
) "ORA GETTERO' I DADI":GOTO 130
```

```
110 IF A=2 THEN PRINT:PRINT:PRINT TAB(5)
 "PREMI RETURN PER GETTARE I DADI"
120 GET A$:IF A$="" THEN 120
130 PRINT:FOR DD=1 TO 1500:NEXT
140 FOR C=1 TO 2
150 PRINT TAB(11) "GETTANDO IL DADO";C
160 FOR DD=1 TO 1500:NEXT
170 D(C)=INT(6*RND(1))+1
180 PRINT TAB(11) "VIEN FUORI";D(C)
185 GOSUB 600
190 FOR DD=1 TO 1500:NEXT
200 PRINT
210 NEXT C
220 IF D(1)<>D(2) THEN PRINT:PRINT TAB(7
) "QUESTE GETTATE NON CONTANO"
230 IF D(1)=D(2) THEN B(A)=B(A)+3*D(1)
235 IF D(1)=D(2) THEN PRINT:PRINT TAB(10
) "LE GETTATE ERANO";D(1);"E";D(2)
240 FOR DD=1 TO 1500:NEXT
300 NEXT A

310 R=R+1:PRINT "♥":GOTO 45:REM CANCELLA
 LO SCHERMO
400 PRINT "♥":POKE 53281,0
410 PRINT TAB(12) "■■■■■■ IL VINCITORE E
' ":REM GIALLO CURSORE GIU'
420 IF B(2)>B(1) THEN PRINT TAB(11) "■■■
■ L'UOMO !!":REM CURSORE GIU'
430 IF B(1)>B(2) THEN PRINT TAB(11) "■■■
■ IL COMPUTER !!"
500 GOTO 500
600 POKE 54296,10
610 POKE 54277,9
620 POKE 54273,17:POKE 54272,30
630 POKE 54276,33
640 FOR DD=1 TO 100:NEXT
650 POKE 54276,0:POKE 54277,0:POKE 54278
,0
660 RETURN
670 POKE 54276,0:POKE 54277,0
700 RETURN
```

```
>>> GIOCO DI DADI <<<

     ROUND NO. 1

     COMPUTER : 0

     UOMO    :    0

  ORA GETTERO' I DADI

   GETTANDO IL DADO 1
   VIEN FUORI 5

   GETTANDO IL DADO 2
   VIEN FUORI 4

QUESTE GETTATE NON CONTANO


>>> GIOCO DI DADI <<<

     ROUND NO. 1

     COMPUTER : 0

     UOMO    :    0

PREMI RETURN PER GETTARE I DADI

   GETTANDO IL DADO 1
   VIEN FUORI 6

   GETTANDO IL DADO 2
   VIEN FUORI 4

QUESTE GETTATE NON CONTANO
```

```
>>> GIOCO DI DADI <<<

ROUND NO. 2

COMPUTER : 0

UOMO   :   0

ORA GETTERO' I DADI

GETTANDO IL DADO 1
VIEN FUORI 6

GETTANDO IL DADO 2
VIEN FUORI 6

LE GETTATE ERANO 6 E 6


>>> GIOCO DI DADI <<<

ROUND NO. 2

COMPUTER : 18

UOMO   :   0

PREMI RETURN PER GETTARE I DADI

GETTANDO IL DADO 1
VIEN FUORI 2

GETTANDO IL DADO 2
VIEN FUORI 6

QUESTE GETTATE NON CONTANO
```

```
        >>> GIOCO DI DADI <<<

              ROUND NO. 3

              COMPUTER : 18

              UOMO    :    0

         ORA GETTERO' I DADI

          GETTANDO IL DADO 1
          VIEN FUORI 5

          GETTANDO IL DADO 2
          VIEN FUORI 3

      QUESTE GETTATE NON CONTANO


        >>> GIOCO DI DADI <<<

              ROUND NO. 3

              COMPUTER : 18

              UOMO    :    0

    PREMI RETURN PER GETTARE I DADI

          GETTANDO IL DADO 1
          VIEN FUORI 5

          GETTANDO IL DADO 2
          VIEN FUORI 5

         LE GETTATE ERANO 5 E 5
```

```
>>> GIOCO DI DADI <<<

ROUND NO. 4

COMPUTER : 18

UOMO    :   15


ORA GETTERO' I DADI

GETTANDO IL DADO 1
VIEN FUORI 6

GETTANDO IL DADO 2
VIEN FUORI 3


QUESTE GETTATE NON CONTANO
```

A P P E N D I C E

SIMBOLI MATEMATICI

Ecco un sommario dei piu' importanti simboli matematici del COMMODORE 64:

Simboli abituali	Simboli del computer
+ (piu')	+
- (meno)	-
x (per)	*
: (diviso)	/
m elevato n (elevamento a potenza)	m↑n

Le funzioni matematiche includono:

Computer	Significato
ATN	Arcotangente
SIN	Seno
COS	Coseno
TAN	Tangente
INT	Riduzione al piu' piccolo numero intero
SGN	Segno (riporta -1 se negativo, 0 se zero, 1 se positivo)
ABS	Riporta il numero senza segno (ABS(-5) da' 5)
SQR	Radice quadrata

COME PRENDERSI CURA DEI DISCHI

Ci sono alcune regole da osservare nella cura dei dischi, che potranno assicurarti il loro massimo rendimento. Cio' ridurra' anche la possibilita' di perdere importanti dati a causa del danneggiamento dei dischi.

Se succede qualcosa al disco, cioe' se ti si dovesse piegare o se qualcosa si rovesciasse sulla copertina di cartoncino (e non sul disco stesso), ripara il danno nel miglior modo possibile, e poi fai immediatamente una copia del disco. Quindi scarta l'originale.

Se inserisci nel drive un disco rovinato rischi di danneggiare anche la testina di lettura/scrittura.

Non toccare la parte del disco che si vede attraverso l'apertura della copertura esterna, e non poggiare niente sul disco o sul cartoncino di copertura.

Il magnetismo puo' rovinare i dati presenti su disco. Se possiedi qualcosa che puo', o potrebbe essere, magnetizzato, tienilo ben distante dal tuo disco.

I dischi sono molto sensibili. Essi non sopportano molto il caldo, cosi' lasciarli in macchina per molto tempo puo' causare gravi problemi (il migliore intervallo di temperatura per conservare i dischi va dai 50 F ai 125 F ossia da 10 C a 51 C;se il disco resta esposto a temperature maggiori o minori di quelle indicate, lascialo almeno una mezz'ora fuori, prima di inserirlo nel drive). Non dovrai mai piegare i dischi, o usare copertine rovinate, o nastri adesivi o graffette.

Se vuoi scrivere sulla fascetta della copertina di cartone, prima tira fuori il disco. Scrivi sull'etichetta che deve essere incollata sul disco stesso, prima di incollarla. Se dovrai aggiungere qualcos'altro in seguito, scrivi delicatamente, e con una penna dalla punta morbida. Non usare comunque una penna a sfera o una penna appuntita.

GLOSSARIO DI PAROLE DEL COMPUTER

Address- un numero che si riferisce ad una locazione, generalmente nella memoria del computer, dove l'informazione e' stata immagazzinata.

Algorithm- la sequenza di passi usata per risolvere un problema.

Alphanumeric- generalmente usato per descrivere una tastiera, vuol dire che la tastiera possiede tasti numerici e alfabetici.

APL- Automatic Programming Language, un linguaggio creato da Iverson, che contiene un elevato numero di operatori e strutture di dati. Usa dei caratteri non standard.

Application software- questi sono programmi utilizzati per compiti specifici, come ad esempio word processing.

ASCII- American Standard Code for Information Interchange. Questo e' un codice quasi universale per lettere, numeri e simboli, che ha un numero tra zero e 255 assegnato ad ognuno di questi, ad esempio il 65 per la lettera A.

Assembler- Questo e' un programma che converte un altro programma scritto nel linguaggio assembly (che e' un programma del computer in cui una singola istruzione, come per esempio ADD, viene convertita in una singola istruzione per il computer) nel linguaggio usato direttamente dal computer.

BASIC- sta per Beginner's All-purpose Symbolic Instruction Code, il linguaggio piu' comune usato sui microcomputer. E' facile da imparare.

Batch- un gruppo di transazioni che deve essere elaborato dal computer, senza interruzioni da parte di un operatore.

Baud- misura di velocita' di trasferimento dei dati. Generalmente sta per il numero di bits al secondo.

Benchmark- un test usato per misurare alcuni aspetti della performance di un computer, che puo' essere paragonato al risultato di esecuzione di un test simile su un computer differente.

Binary- sistema numerico con due soli simboli, 0 e 1 (opposto al sistema decimale ordinario di dieci simboli). Il tuo computer "pensa" in binary.

Boolean algebra- l'algebra decisionale e logica, sviluppata da George Boole ed e' alla base del sistema decisionale del computer.

Bootstrap- un programma, fatto girare nel computer appena acceso, che dice al computer di accettare e comprendere altri programmi.

Buffer- magazzino che contiene input da una periferica, come ad esempio la tastiera, e che viene poi trasferito ad una velocita' dettata dal computer.

Bug- errore nel programma.

Bus- un gruppo di connessioni elettriche usato per congiungere un computer con una periferica o un altro computer.

Byte- il piu' piccolo gruppo di bits che compone una parola computer. Generalmente un computer e' descritto come avente 8 bits o 16 bits, cioe' che una parola consiste in una combinazione di 8 o 16 zero o uno.

CPU- (Central Processing Unit) il cuore del computer che svolge funzioni aritmetiche, logiche e di controllo.

Character code- il numero in ASCII che si riferisce ad un particolare simbolo, come ad esempio 32 per uno spazio.

COBOL- COmmon Business Oriented Language, linguaggio standard prossimo all'inglese, usato principalmente per affari.

Compiler- un programma che traduce un programma scritto in un linguaggio umano, in linguaggio macchina compreso direttamente dal computer.

Concatenate- aggiungere (aggiungere due stringhe insieme si dice concatenazione).

CP/M- Control Program Microcomputer, un sistema di disco quasi universale creato e diffuso da Digital Research, California.

Data- termine generale per un'informazione eseguita dal computer.

Database- un gruppo di dati, organizzato per permettere un rapido accesso al computer.

Debug- rimuovere errori da un programma.

Disk- un sistema di immagazzinamento magnetico (altrimenti descritto come "hard disk", "floppy disk", o anche "floppy") usato per immagazzinare informazioni del computer e programmi. Il disco assomiglia, pressappoco, ad un 45 giri, ed ha generalmente otto, cinque e 1/4, o tre pollici di diametro. Esistono anche microdischi, usati per alcuni sistemi.

Documentation- istruzioni scritte e spiegazioni che accompagnano un programma.

DOS- Disk Operating System, il programma versatile che permette che il computer controlli un sistema a disco.

Dot-matrix printer- una stampante che forma lettere e simboli con un gruppo di dots, usati generalmente su di una griglia di 8x8 o 7x5.

Double-density- aggettivo usato per descrivere dischi registrati, che usa una tecnica speciale che, come suggerisce il nome, raddoppia l'ammontare di immagazzinamento che il disco puo' fornire.

Dynamic memory- memoria del computer che richiede ricambi costanti.

EPROM- Erasable Programmable Read Only Memory, una periferica che contiene informazioni del computer in forma semipermanente, che richiede un'esposizione ai raggi ultravioletti per cancellare i suoi contenuti.

Error messages- informazione dettata dal computer all'operatore che consiste a volte in soli numeri o poche lettere, ma generalmente in una frase, che sottolinea un errore di programma o di operazione che ha fatto in modo che il computer si fermasse in fase di esecuzione.

Field- un gruppo distinto di caratteri, come ad esempio un codice identificante, un nome o una data, un field fa generalmente parte di un disco.

File- un gruppo di records che vengono eseguiti insieme.

Firmware- componenti solidi di un computer sono spesso chiamati hardware; i programmi in una forma leggibile dalla macchina su disco o cassetta, sono chiamati software; e i programmi registrti su circuito sono chiamati firmware. Questi possono essere alterati, in alcune circostanze, dai software.

Flag- questo e' un indicatore all'interno di un programma, con un valore, che da' informazioni riguardanti una particolare condizione.

Floppy disk- vedi disk.

Flowchart- e' generalmente scritto prima di qualsiasi altra riga di un programma che si ha intenzione di inserire nel computer, che fa corrispondere varie forme geometriche a determinate fasi nello sviluppo del programma.

FORTRAN- un linguaggio ad alto livello usato generalmente per problemi scientifici.

Gate- un componente del computer che prende decisioni, permettendo al circuito di muoversi in una direzione o in un altra, a seconda delle indicazioni delle condizioni.

GIGO- Garbage In Garbage Out, suggerisce che se vengono inseriti dati erronei nel computer, il risultato del programma sara' anch'esso errato.

Global- Un gruppo di condizioni che influenza un intero programma, viene chiamato global al contrario di local.

Graphic- termine per qualsiasi output del computer non alfanumerico o simbolico.

Hardcopy- informazione scritta su carta da una stampante.

Hardware- vedi firmware e software.

Hexadecimal- un sistema numerico basato sul numero 16, opposto al sistema ordinario che e' basato sul numero dieci.

Hex pad- una tastiera che e' usata per inserire numeri esadecimali.

High-level languages- linguaggi prossimi all'inglese. I low-level linguaggi sono prossimi al linguaggio macchina. Siccome gli high-level linguaggi devono essere compilati in una forma piu' facilmente comprensibile dal computer prima di essere eseguiti, essi girano molto piu' lentamente dei low-level linguaggi.

Input- informazione immessa nel programma durante l'esecuzione.

I/O- Input/Output, periferiche che il computer usa per comunicare con il mondo esterno.

Instruction- elemento del codice del programma che dice al computer di svolgere un compito specifico. Un'istruzione nel linguaggio assembly, per esempio, e' ADD, che dice al computer di svolgere un'addizione.

Interpreter- converte il programma ad alto livello in una forma comprensibile per il computer.

Joystick- una periferica che inserisce segnali nel computer, relativi alla posizione che il joystick sta occupando, generalmente usata in programmi di giochi.

Kilobyte- unita' di misura di linguaggio; un Kbyte equivale a 1024 bytes.

Line printer- una stampante che stampa, nello stesso tempo, una riga completa di caratteri.

Low-level language- linguaggio simile al linguaggio macchina (vedi high-level language).

Machine language- si trova un gradino sotto il low-level language, ed e' compreso direttamente dal computer.

Mainframe- il termine per i computer giganti come l'IBM 370. I computer vengono anche classificati in minicomputers e microcomputers.

Memory- le periferiche usate da un computer per immagazzinare informazioni e programmi durante la loro esecuzione, e per le istruzioni poste all'interno di un computer che dicono come svolgere le richieste di un programma. Ci sono principalmente due tipi di memoria (vedi RAM e ROM).

Microprocessor- il chip che giace nel cuore del computer.
Modem- sta per MOdulator-DEModulator ed e' una periferica che permette al computer di comunicare con un'altra via telefono.

Monitor- (a) uno schermo del televisore usato come unita' di display del computer che non contiene l'apparato di sintonia; (b) l'informazione in un computer che permette ad esso di capire ed eseguire le istruzioni del programma.

Motherboard- un'unita', generalmente esterna, che possiede degli slots che permettono di inserire nel computer dei circuiti aggiuntivi non contenuti nella macchina standard.

Mouse- un'unita' di controllo, piu' piccola di un pacchetto di sigarette, che ruota sulla scrivania muovendo un cursore sullo schermo parallelamente, per prendere decisioni in un programma. I mouses lavorano anche utilizzando le loro ruote, oppure leggendo una griglia sulla superficie su cui si stanno muovendo.

Network- un gruppo di computer che lavorano insieme.

Numeric pad- una periferica utilizzata per inserire

informazioni numeriche in un computer, simile ad un calcolatore.

Octal- un sistema numerico basato su 8.

On-line- periferica sotto il diretto controllo del computer.

Operating system- questo e' il programma principale o una serie di programmi nel computer, che controlla le operazioni del computer, come per esempio chiamare le routine o assegnare priorita'.

Output- qualsiasi dato prodotto dal computer durante l'esecuzione, sia sullo schermo sia sulla stampante, o usato internamente.

Pascal- linguaggio ad alto livello creato da Niklaus Wirth, usato principalmente per programmi di disciplina e di struttura.

Port- un output o input "buco" nel computer, attraverso il quale viene trasferito un dato.

Program- serie di istruzioni che il computer segue per svolgere un compito predeterminato.

Pilot- linguaggio ad alto livello, generalmente usato nei programmi educativi.
RAM- Random Access Memory, memoria che ritiene il programma che si sta svolgendo. I contenuti della RAM possono essere cambiati, al contrario di quelli della ROM (Read Only Memory).

Real-time- quando un'operazione nel computer si svolge in concomitanza con il tempo reale, viene detta svolgentesi in real-time. Molti giochi richiedono reazioni in tempo reale.

Refresh- i contenuti delle memorie dinamiche devono ricevere dell'energia periodica per mantenere i propri

contenuti. Il segnale che ricorda alla memoria i suoi contenuti e' chiamato refresh signal.

Register- locazione nella memoria del computer che ritiene i dati.

Reset- segnale che riporta il computer al punto in cui si trovava al momento dell'accensione.

ROM- vedi RAM.

RS-232- un'interfaccia standard serial che connette un modem ed equipaggiamenti terminali associati ad un computer.

S-100 bus- e' un'altra interfaccia standard fatta di 100 righe di comunicazione parallele, usate per connettere circuiti nei microcomputers.

SNOBOL- un linguaggio ad alto livello, sviluppato dai laboratori Bell, che usa manipolazioni di stringhe.

Software- programma seguito dal computer (vedi firmware).

Stack- il punto finale di una serie di operazioni.

Subroutine- una serie di istruzioni chiamate in un programma per un certo numero di volte.

Syntax- sintassi.

Systems software- sezioni di un codice che svolgono doveri amministrativi, o assistono con la scrittura di altri programmi, ma che attualmente non svolgono il compito finale del computer.

Thermal printer- una periferica che stampa l'output dal computer su una carta sensibile al caldo. Sebbene le stampanti termiche siano piu' silenziose, l'output non risulta facilmente leggibile e nemmeno la carta usata, facile da immagazzinare.

Time sharing- questo termine e' usato per riferirsi a un grosso numero di operatori, su terminali indipendenti, facenti uso dello stesso computer, che divide il tempo tra gli operatori, in modo tale che ad ognuno di essi paia avere un computer a completa disposizione.

Turnkey system- sistema del computer pronto da far girare appena emesso che necessita soltanto di un "giro di tasto" perfarlo funzionare.

Volatile memory- periferica che perde i suoi contenuti quando viene a mancare l'energia (vedi memory, refresh, ROM e RAM).

Word processor- un programma che rende il computer un'intelligente macchina da scrivere, con uno svariato numero di correzioni e caratteristiche di stampa.

Tim Hartnell, uno dei più prolifici ed esperti autori di computer, ha raccolto in questo volume, oltre 50 esempi applicativi di routines e programmi di giochi, matematica, utilità e musica i più interessanti dei quali sono riportati su cassetta.
Oltre ai programmi, troverete decine e decine di consigli che vi permetteranno di avvicinarvi in modo divertente all'affascinante mondo dell'informatica.
In questo libro infatti, sono esposte le cose fondamentali per l'uso e la programmazione in BASIC del COMMODORE 64, scritte in forma chiara ed esauriente.
In più vi potrete avvalere della **cassetta software** per un più rapido apprendimento delle tecniche di programmazione.
Dopo aver letto questo libro, sarete in grado di impressionare gli amici per la vostra abilità, insegnare ai vostri figli e creare da soli, programmi in BASIC di utilità e svago.

Cod. 9151

L. 25.000
prezzo comprensivo di cassetta